E901757

AFATL-TR-87-54

2

Image Processing Language, Phase I

Charles R Giardina
Edward R Dougherty

THE SINGER COMPANY
ELECTRONIC SYSTEMS DIVISION
164 TOTOWA ROAD
WAYNE, NJ 07474-0975

MAY 1988

FINAL REPORT FOR PERIOD SEPTEMBER 1984-SEPTEMBER 1985

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

AIR FORCE ARMAMENT LABORATORY
Air Force Systems Command ▮ United States Air Force ▮ Eglin Air Force Base, Florida

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility nor any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service (NTIS), where it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

Donald C Daniel

DONALD C. DANIEL
Chief, Advanced Guidance Division

Please do not request copies of this report from the Air Force Armament Laboratory. Copies may be obtained from DTIC. Address your request for additional copies to:

Defense Technical Information Center
Cameron Station
Alexandria, VA 22304-6145

If your address has changed, if you wish to be removed from our mailing list, or if your organization no longer employs the addressee, please notify AFATL/AGS, Eglin AFB FL 32542-5434, to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for public release.
Distribution is unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S)

5. MONITORING ORGANIZATION REPORT NUMBER(S)
AFATL-TR-87-54

6a. NAME OF PERFORMING ORGANIZATION
Singer, Electronic
Systems Division

6b. OFFICE SYMBOL *(If applicable)*
CN 975

7a. NAME OF MONITORING ORGANIZATION
Air-to-Surface Guidance Branch (AGS)
Advanced Guidance Division (AG)

6c. ADDRESS *(City, State, and ZIP Code)*
164 Totowa Road
Wayne, N.J. 07474-0975

7b. ADDRESS *(City, State, and ZIP Code)*
Air Force Armanent Laboratory
Eglin AFB, FL 32542-5434

8a. NAME OF FUNDING / SPONSORING ORGANIZATION
AFATL/AGS & DARPA/TTO

8b. OFFICE SYMBOL *(If applicable)*

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
F08635-84-C-0296

8c. ADDRESS *(City, State, and ZIP Code)*

AFATL/AGS
Eglin AFB FL
32542-5434

DARPA/TTO
1400 Wilson Blvd
Arlington, VA 22209

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT CESSION NO.
62602F	2068	06	52

11. TITLE *(Include Security Classification)*
Image Processing Language, Phase I

12. PERSONAL AUTHOR(S)
Charles R. Giardina and Edward R. Dougherty

13a. TYPE OF REPORT
Final

13b. TIME COVERED
FROM Sept 84 TO Sept85

14. DATE OF REPORT *(Year, Month, Day)*
May 1988

15. PAGE COUNT
180

16. SUPPLEMENTARY NOTATION
Availability of this report is specified on verso of front cover.

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
16	04	
17	07	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Image Algebra
Image Processing
Pattern Recognition
Elemental Operators
Edge Detector
Image Algorithm

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

The demand for image processing activities in military, industrial, and academic communities has greatly increased and has resulted in a deluge of different image architectures, operations, and notation systems. A standardized, mathematically rigorous, efficient algebraic system designed specifically for image manipulation does not exist. This report, Image Algebra, Phase I, develops a standardized mathematical structure for the basis of image processing algorithms and techniques. The report presents the development of ten underlying basic or elemental operators from the perspective of both mathematical and machine implementations. Seven elemental operators are familiar: addition, multiplication, division, translation, rotation, maximum and reflection. Two operators are projections for extracting image from the domain and range. The tenth operator, the existential operator, allows the formation of an image. The elemental operators are in-turn used in image algorithm oriented macro-operators and spanning capabilities.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
□ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. □ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL
BRUCE T. BULLER

22b. TELEPHONE *(Include Area Code)*
904-882-2968

22c. OFFICE SYMBOL
AFATL/AGS

Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

PREFACE

This program was conducted by the Singer Company, Electronic Systems Division, 164 Totowa Road, Wayne, New Jersey 07474-0975, under Contract Number F08635-84-C-0296 with the Air Force Armament Laboratory, Eglin Air Force Base, Florida 32542-5434. Mr Charles R. Giardina was the principal investigator, and Mr Edward R. Dougherty was the consultant. Mr Neal Urquhart (AFATL/AGS) managed the program for the Armament Laboratory. The program was conducted during the period from September 1984 to September 1985.

The authors wish to extend appreciation to both the Air Force Armament Laboratory and the Defense Advanced Research Project Agency for sponsoring the image processing program. In particular, the efforts of Mr Neal Urquhart (AFATL/AGS) deserve notice. His recognition of the need for an image algebra has brought the project about, and his counsel has been essential for providing direction.

TABLE OF CONTENTS

TABLE OF CONTENTS (CONCLUDED)

LIST OF FIGURES

LIST OF TABLES

SECTION I

INTRODUCTION

1. BACKGROUND

As is the case with any newly developing technological area, image processing has tended to evolve in an ad hoc manner. There has been little or no effort at standardization of definitions, structural notation, algorithm specification, terminology and methodology. An important first step in the direction of standardization is the development of a uniform underlying mathematical structure for the expression of image processing algorithms.

If one surveys the literature, he at once recognizes the disparate manner in which algorithms are specified. It is extremely difficult to recognize procedures which are essentially the same but are being presented by two different authors. Moreover, the lack of any underlying set of fundamental image processing operations (at a low level) makes optimization and the reduction of complexity essentially impossible. It is as if one attempts to proceed through algebra and calculus without any understanding of the basic operations of arithmetic and the laws pertaining to these operations.

Because of the present chaotic state of image processing algorithm specification, the need for a study of the underlying mathematical operations is apparent. The problem is threefold:

a. The criteria that an underlying set of operations must satisfy must be delineated.

b. A suitable fundamental collection of low level operators must be found.

c. The mathematical structure of the algebra based upon the fundamental operators must be investigated and the entire structure must be placed within the appropriate mathematical framework. It is the intent of the current effort to accomplish the aforementioned tasks.

2. THE PROJECT

This report represents the culmination of approximately one year's effort to find an imaging algebra. Though much developmental work remains to be done, the skeleton of a satisfactory structure appears to have been found. It is the intent of this report to define, explain, illustrate and demonstrate the capabilities of the proposed algebraic structure.

Several germane attributes of the proposed structure are:

a. mathematical and computer implementation of the basis

b. range and domain induced basic operators

c. basic projection operators

d. spanning capability of the basis

e. image macro operators

f. ordered basic and macro operators by complexity.

The attributes represent, in compact form, those properties that were identified in the pro}ect as being critical to the eventual success of the system.

SECTION II

FUNDAMENTAL OPERATORS IN THE IMAGE ALGEBRA

1. BASIS

A set of criteria that a collection of elemental operations should satisfy in order to qualify as a candidate for a basis that will underlie the development of an image processing algebra has been articulated. The primary role of a basis is to serve as a construct to categorize thinking at a certain level. Once such a categorization is given, uniformity of structure results. The ability to communicate is enhanced and the development of linguistic models is made possible. Moreover, there is no loss of freedom, since, as the category of thinking gets broader, the existing definition of the basis can be concomitantly broadened to accommodate the novel concepts. Should a particular field encompass several seemingly disjoint transformation types, as does image processing, a level approach to basis construction can be taken. Each level may have its own mini-basis and the basis of the entire collection of operations may be taken as a union of the individual mini-bases. In terms of an image processing algebra, there are various levels to be considered. Therefore, it is appropriate to take a modular approach.

Essentially, there are several criteria the overall basis should satisfy. It should be representable. Important image operations should be definable under function composition using the elemental operations of the basis. This is the so-called spanning capability of the basis. Those algorithms for which there exists a basis representation will be part of the resulting system. If the basis is to have good spanning characteristics, these representable algorithms must form a class which contains the vast majority of the existing procedures.

The addition of new and important operations might require an expansion of the basis.

A second criterion the basis should satisfy is that of manipulability. It should be convenient to use in that high level functions and macro-functions are for the most part readily obtainable from the basis elements. It should be modular and also provide views at various levels. There should also be a general simplicity so that the underlying operations are easily visualized and understood.

Next, the basis should be efficient. The desire is for elementary operations, though not necessarily the most elementary. The overall basis should be space-time efficient and thereby provide a pragmatically functional system. It should support a collection of macro-operators from which the varied imaging operations can be expressed. Although the basic operators may not be independent of one another in a strict mathematical sense, needless redundancy must be avoided. In a sense, this last criterion embodies the essential thinking of the Image Processing Language Program: The ultimate goal is not a system which is minimal from a strictly logical perspective, but one that provides a structured framework for the practical expression of useful algorithms.

2. IMAGE ALGEBRA CRITERIA

A set of criteria that a mathematical structure should satisfy in order to qualify as a candidate for an image processing algebra has been ascertained. From a rigorously logical point of view, the image algebra itself is mainly determined by the choice of basis. Nevertheless, it is useful to specifically articulate those properties which are desirable for the algebraic system as a whole. The image algebra must satisfy certain heuristic conditions in order to serve as the supporting structure for image processing. There are, in fact, many different bases which lead to the same algebra. Therefore, while there is interplay between the basic criteria and the algebraic criteria, they are to some extent exclusive. As a result, the desirable properties for an imaging algebra need to be treated separately.

The algebra must be effective and efficient. It is effective to the extent that it enables autonomous target detection and classification algorithms to be represented and developed.

Necessary to a pratical effectiveness is simplicity and clarity; the algebra must be accessible to those who desire to use it. Its efficiency depends upon the extent to which it allows for algorithms to be developed in a favorable fashion with respect to cost and resources. It must allow for the ready exploitation of the parallelism which is inherent in so many imaging algorithms.

The algebra should unify many typed criteria. It should serve as a vehicle for bringing together the many diverse areas of image processing through the utilization of precise specifications.

The imaging algebra should be at once expandable and robust. Expandability requires that there be a capability to delete, insert or modify operators. Robustness requires that the schema should have little or no variation with changes in operators, types or constraints. Moreover, the formalism must be adaptive to changes due to advances in mathematics, in the characteristics of imaging sensors and in the architecture of processors and memory elements.

The imaging algebra structure should support object oriented design. This requires it to be programmably transportable. It should be an easy task to go from operators in the algebra into code for most machines. The framework should also support a disciplined programming style with various levels of abstraction. It should lead to brevity, clarity, modularity and concinnity.

Certainly all of the preceding conditions cannot be satisfied in their entirety. Nonetheless, they can serve as guidelines to which the construction of a useful and comprehensive imaging algebra might aspire.

3. THE ELEMENTAL OPERATORS

The most essential property of any set of fundamental operators, or basis, in an imaging algebra is its spanning capability, that is, the ability to serve as a set of elemental operations from which image processing algorithms can be constructed. Without a good spanning capacity, a basis, and hence the resulting imaging algebra, would fall short, no matter how excellent its other characteristics. The proposed basis has the desired spanning capability while at the same time being composed of operations which are both simple and natural.

In order to appreciate the power and simplicity of the proposed basis, it is important to recognize that the construction of a satisfactory imaging algebra requires at the outset the exposure of the structures that underlie the operational specification of image processing algorithms. As with most mathematics, these primitive structures tend to be quite simple. In general, the end product of mathematical reasoning can be elaborate and difficult for the non-expert to penetrate; however, the premises from which the reasoning begins are usually not overly complex. In the case of the proposed imaging algebra, its structure must allow for the development of most current and (hopefully) future imaging transformations.

These may ultimately prove to be of a high order of complexity; nevertheless, they must spring from some low level of primitives. These, in turn, will be a by-product of the supporting mathematical structures upon which the operations are based.

While the preceding remarks tend to be philosophical in nature, once the structural particulars of imaging algorithms are discovered, they lead directly to the proposed basis. A digital image defined herein is a partial function on ZxZ into the reals, that is, it is a function whose domain is a subset of ZxZ and whose codomain is the real number system R. The domain is the extent and codomain is the grey-scale of an image. The set of all images will be denoted by X. It is mathematically natural to look within the structures of ZxZ and R to find the primitive operations of image processing. Both ZxZ and R are extremely rich and well-studied mathematical entities. Each has an extensive structure from which to draw. The proposed basis was developed by drawing upon those domain (ZxZ) and codomain (R) structural properties which play a role in digital image procesing. As occurs through mathematics, these lead at once to corresponding properties (or, in this case, operations) within the new structure which they together induce. Therefore, there naturally arises a set of domain induced (from ZxZ) operations and a set of codomain induced (from R) operations. In a sense, one might say that these are there to be found. For a succesful image algebra, one needs to select those operations which are required for the convenient representation of digital imaging transformations.

It must be understood that while the preceding comments provide a natural approach to the basis selection problem, they do not provide a deterministic methodology. Pragmatic modelling decisions must be made. Not only does one have to search the literature to see what is going on, one must recognize that different images can have different domains within ZxZ. The decision as to how to proceed when, for example, one desires to add two images with different domains must be made in a heuristic manner. In making such decisions for the proposed basis, an attempt has been made to define the elemental operations in a way which reflects the manner in which the induced operations are most used in practice. Fortunately, it turns out that in every instance that has come to attention, other natural choices for the induced elemental operations are derivable as terms in the algebra or as Macro-Operators from the chosen basis set. These macro-operators are given in a later section along with a rigorous discussion of the inducement process.

TABLE 1. FUNDAMENTAL OPERATORS IN IMAGE ALGEBRA

I	Addition	$\oplus$
II	Multiplication	$\odot$
III	Maximum	$\circledvee$
IV	Division	$\oplus$
V	Translation	T
VI	Rotation	N
VII	Reflection	D
VIII	Domain Extractor	K
IX	Parameter Extractor	G
X	Existential Operator	E

4. SPECIFICATION OF OPERATORS

The first four fundamental operators to be introduced are range induced, and they include addition, multiplication, maximum and division. The next three operations are translation, rotation, and reflection. They also take digital images into digital images; however, they are domain induced. The final three operations in the basis do not take images into images. They include the domain extraction operation, which takes an image and returns a subset of $Z \times Z$, the parameter extraction operation, which maps an image into the reals, and the existential operation, which is used in creating an image.

An image is a real-valued mapping defined on a subset of the integral lattice $Z \times Z$. Symbolically, an image is a mapping $f: A \rightarrow R$, where $A \subset Z \times Z$. We also employ the customary notation R^A for the class of all such mappings, f from A into R. Note that for the null set $\emptyset \subset Z \times Z$, we obtain the so-called null image. It has an empty domain. As for the collection of all images, we denote this class by X and

$$X = \bigcup_{A \subset Z \times Z} R^A \quad .$$

Insofar as a particular grey value of an image $f \in R^A$ is concerned, this is denoted by $f(i,j)$, where $(i,j) \in A \subset Z \times Z$. The first element of the pair, i, gives the position on the x-axis, while the second, j, gives the position on the y-axis.

a. Addition (Range Induced). Since each pixel in the domain of an image has a grey value which is an element of R, the real number system, and since there is a natural addition (+) in R, there is an induced addition defined as a binary operation on images. This image addition is denoted by $\oplus$ and is a basis operation. For each pixel in the intersection of the input domains, the output image has the arithmetic sum of the input grey values at that pixel. For a pixel which lies in one of the input domains but not both, the decision has been made to leave its grey value unchanged. A similar decision has been made regarding the mu tiplication operator and the maximum operator, each of which will be considered in turn. We define $\oplus : X \times X \rightarrow X$ as follows:

If the domain of f is A and the domain of g is B, then the domain of $f \oplus g$ is $A \cup B$ and

$$(f \oplus g) = \begin{cases} f(x,y) & (x,y) \in A - B \\ g(x,y) & (x,y) \in B - A \\ f(x,y) + g(x,y) & (x,y) \in A \cap B \end{cases}$$

b. **Multiplication (Range Induced).** Similar reasoning as given in the addition operation is applied to the natural multiplication $(\cdot)$ in R. The result is a pixelwise induced multiplication operation on pairs of input images. For a pixel in the intersection of the domains of the input images, the corresponding grey values are multiplied. On the other hand, the grey value of a pixel which lies in only one domain of the input images is left unchanged.

Hence we define the binary operator

$\odot : X \times X \rightarrow X$, where the operands are images and the output is also an image, as follows:

Let the domain of f be A and the domain of g be B. Then the domain of $f \odot g$ is $A \cup B$ and

$$(f \odot g)(x,y) = \begin{cases} f(x,y) & (x,y) \in A - B \\ g(x,y) & (x,y) \in B - A \\ f(x,y) \cdot g(x,y) & (x,y) \in A \cap B \end{cases}$$

c. **Maximum (Range Induced).** Given two real numbers in R there is a natural order operation called maximum. Simply stated, for two real numbers y and z, $y \vee z$ is either y, z or their common value, depending respectively upon whether y is greater, z is greater or they are equal. This naturally induces a pixelwise maximum on the intersection of two input domains. Once again, the heuristic determination has been made to leave the input images unaltered off the intersection. The operation is denoted by $\ovee$.

We define $\ovee : X \times X \rightarrow X$, where the domain of f is A and the domain of g is B, then the domain of $f \ovee g$ is $A \cup B$ and

$$(f \ovee g)(x,y) = \begin{cases} f(x,y) & (x,y) \in A - B \\ g(x,y) & (x,y) \in B - A \\ f(x,y) \vee g(x,y) & (x,y) \in A \cap B \end{cases}$$

d. Division (Range Induced). Each grey value z which is not zero has a reciprocal grey value 1/z. Hence there is a natural image operation, called division, which replaces each nonzero grey value by its reciprocal. It is denoted by $\oplus$. Since in R the reciprocal of zero is undefined, it has been decided that the division operation should leave the output image undefined at any pixel for which the input image has grey value zero.

Consequently, $\oplus : X \rightarrow X$, where if there is a zero pixel in the input image then the output image has a smaller domain than the input.

Specifically, $(\oplus f)(x,y) = 1/f(x,y)$ if $f(x,y) \neq 0$ and is undefined if $f(x,y) = 0$. When the division is preceded by a multiplication operator $\odot$, we shall omit $\odot$.

e. Translation (Domain Induced). Given a position vector in a two dimensional space, denoted by (m,n), the vector addition between (m,n) and another vector (i,j) yields a new position vector (m+i,n+j). Geometrically, the original position is moved over (x direction) i units to the right and up (y direction) j units. This position operation induced the translation operation on images. The elemental operator T moves an image over and up, while leaving grey values unchanged. Notationally, T(f,i,j) or f_{ij} is used to indicate the image obtained by moving it over units and up j units. It is this domain induced operator T which has provea to be invaluable in the exploitation of the natural parallelism which exists in many imaging operations. We define

T: $X \times Z \times Z \rightarrow X$, where T is the trinary operator defined by:

$$(T(f,i,j))\ (x,y) = f(x-i,y-j)$$

f. Ninety Degree Rotation (Domain Induced). A set of ordered pairs in the two dimensional lattice ZxZ can be rotated 90° in the counter-clockwise direction. This at once induces a 90° rotation operation N. The grey values of the input images are left unchanged and the image is simply rotated. Consequently: $N : X \rightarrow X$ is $(N(f))\ (x,y) = f(y,-x)$.

g. Diagonal Reflection (Domain Induced). This operation is similar in origin to the 90° rotation, except that the image is flipped out of the page around a 135° line through the origin. This operation is denoted by D. $D : X \rightarrow X$, by $(D(f))\ (x,y) = f(-y,-x)$. Hence D makes row pixels become column pixels (and conversely) by rotating the image 180° out of the page about the -45° axis.

h. Domain Extractor. The domain of an image is a subset of ZxZ. It is natural and convenient to consider the operation K which takes an image and yields a subset of ZxZ, that subset being the domain of the image. Hence $K : X \rightarrow 2^{Z \times Z}$, and for f in R^A, $A \subset Z \times Z$, $K(f) = A$.

i. Parameter Extractor. Each pixel in the domain of an image has a given grey value. It is often necessary to read out that value, which is an element of R, the codomain. It turns out that it is only necessary to assume the ability to extract the grey value at the origin pixel. Others can be found first applying the appropriate translations. This basis operation yields the grey value at the origin pixel for a given input image. The complexity of the rigorous definition results from the desire to have this operator defined even if the grey value at the origin is undefined. In that event, the closest grey value is chosen. This latter stipulation is essentially just a mathematical formality since it is possible to move any grey value to the origin by translation. The operator G extracts the grey value of the pixel which is closest to the origin in Euclidean distance and at the smallest angle from the abscissa.

Hence G: X → R, where

$$G(f) = \begin{cases} f(0,0) & \text{when } f \in R^{A} \text{ and } (0,0) \in A \\ 0 & \text{when } f = \emptyset \\ f(i,j) & \text{otherwise, where } \tan^{-1}(j/i) \text{ is minimized for minimum } i^2 + j^2. \end{cases}$$

j. **Existential Operator.** Somewhat opposite of the domain finding operation and parameter extraction operation is the Existential Operation E. This operator is a binary operator used in manufacturing an image. The inputs of this operator are a grey value t and a subset A of $Z \times Z$. The output is a constant image with domain A such that every pixel in A has grey value t. Such an image is denoted by t_A ; i.e., $t_A(i,j) = t$ if $(i,j) \in A$ and is undefined otherwise. Thus we define $E : R \times 2^{Z \times Z} \rightarrow X$ where $t \in R$ and $A \in 2^{Z \times Z}$, $E(t,A) = t_A$. The aforementioned operations form the proposed basis. Each taken singularly is very simple in its structure. Yet taken as a collection, they possess a powerful spanning capability insofar as the following transformation types are concerned: image to image, image to parameter, and image to set. Both their simplicity and their power result from the inducement methodology which brings them to light.

SECTION III

CONCLUSION

This report represents the culmination of approximately one year's effort to find an imaging algebra. Though much developmental work remains, the skeleton of a satisfactory structure appears to have been found. An algebraic structure in the form of a many sorted algebra has been presented to describe operations used in image processing. This system involves ten fundamental operations. The germane attributes of this structure follow:

a. The ten underlying basic operators are elemental from the perspectives of both mathematics and machine implementation.

b. Seven of the basic operators are either range induced, or domain induced, thereby, rendering them both operationally and structurally familiar.

c. Two of the operators are projections, one extracting the domain of an image and the other allowing the extraction of the range. The tenth operator, the existential operator, allows the formation of an image. These three operators provide for extensive data structure manipulation and for easy movement among the sorts within the image algebra.

d. An already well-developed collection of image algorithm oriented- macro-operators has been developed. Structural evaluation of these macros is in progress. The macros provide a workable vehicle for the transposition of existing image processing algorithms into the algebra.

e. The spanning capability of the proposed basis is extensive; indeed, not one strictly digital algorithm has yet been presented which is not expressable in terms of the basic ten operators.

f. By telescoping the basic operators and the macros, arranged in some sort of order of increasing complexity, it should be possible to develop an image processing language in which the user has access at all levels down to the basic set of ten. The resulting language will make full availability of the inherent parallelism within imaging algorithms.

The preceding attributes of the algebra represent, in compact form, those properties that were identified early in the project as being critical to the eventual success of the system. As a consequence, we believe the project has, to this date, attained or surpassed all of its original goals.

APPENDIX A

PHASE I ACTIVITIES

APPENDIX A

PHASE I ACTIVITIES

1. CONSOLIDATION/CLASSIFICATION/DESCRIPTION

A list has been compiled of existing image processing transforms, image measurement techniques and feature vector analysis techniques. An attempt has been made to include numerous methods occurring in the current literature and in every known text on image processing. The names of over a hundred of the operators in this listing appear in Appendix D of this document.

The transforms, measurement techniques and feature vector analysis techniques in the aforementioned list have been classified according to the nature of their functions. Categories in the classification schema include image enhancement, edge detection, statistical measurement, feature vector analysis, segmentation, image coding, image compression, clustering, data structure labeling, image reconstruction, noise reduction, image texture modeling, thinning, decomposition techniques, classification, geometric parametrization, grey level distribution, filtering/smoothing, normalization, geometric estimation, connectivity criteria, size and shape description and digitalization.

A brief characterization of each transform, measurement technique, and feature vector analysis technique has been prepared. The characterization includes a description of the operator's effectiveness as well as its deficiencies. Where alternative versions or equivalent transforms exist, each version has been identified, often along with specific advantages and disadvantages.

2. IDENTIFICATION/DEFINITION OF ELEMENTAL OPERATIONS

Common elemental operations composing the collected transforms and measurement techniques have been investigated and identified, or defined.

Collections of elemental operations that might serve as a bases to generate the image processing transforms and measurement technques described above have been identified. A description of useful elemental operations will be provided below.

3. IDENTIFICATION/DESCRIPTION/EVALUATION OF MATHEMATICAL STRUCTURES

A set of criteria that a collection of elemental operations should satisfy in order to qualify as a candidate for a basis that will underly the development of an image processing algebra has been articulated. The primary role of a basis is to serve as a construct to categorize thinking at a certain level. Once such a categorization is given, uniformity of structure results. The ability to communicate is enhanced and the development of linguistic models is made possible. Moreover, there is no loss of freedom, since, as the category of thinking gets broader, the existing definition of the basis can be concomitantly broadened to accomodate the novel concepts. Should a particular field encompass several seemingly disjoint transformation types, as does image processing, a level approach to basis construction can be taken. Each level may have its own mini-basis and the basis of the entire collection of operations may be taken as a union of the individual mini-bases. In terms of an image processing algebra, there are various levels to be considered. Therefore, it is appropriate to take a modular approach.

Essentially there are several criteria the overall basis should satisfy. First it should be representable. Important image operations should be definable under function composition using the elemental operations of the basis. This is the so-called spanning capability of the basis. Those algorithms for which there exists a basic representation will be a part of the resulting system. If the basis is to have good spanning characteristics, these representable algorithms must form a class which contains the vast majority of the existing procedures. The addition of new and important operations might require an expansion of the basis.

A second criterion the basis should satisfy is that of manipulability. It should be convenient to use in that high level functions and macro-functions are for the most part readily obtainable from the basis elements. It should be modular and also provide views at various levels. There should also be a general simplicity so that the underlying operations are easily visualized and understood.

Next, the basis should be efficient. The desire is for elementary operations, though not necessarily the most elementary. The overall basis should be space-time efficient and thereby provide a pragmatically functional system. It should support a collection of macro-operators from which the varied imaging operations can be expressed. Although the basic operators may not be independent of one another in a strict mathematical sense, this last criterion embodies the essential thinking of the Image Processing Language Program: the ultimate goal is not a system which is minimal from a strictly logical perspective, but one that provides a structured framework for the practical expression of useful algorithms.

While the preceding criteria tend to be heuristic in nature, the last criterion to be discussed, consistency, is a fundamental logical requirement. It must be impossible to deduce a proposition and its negation utilizing the basis elements. While on the surface the consistency condition is readily understandable, the rigorous verification of such a statement can be most difficult; indeed, the notion of consistency goes to the most profound depths of the foundations of mathematics. In this instance, it will suffice to note that the existence of a real world model can be taken as proof of consistency since it is a fundamental postulate of science that two contradictory propositions concerning the real world cannot be simultaneously maintained. In the case of the imaging algebra there is no problem relating to consistency, since all of the basis operators are induced from either the algebraic structure of the grid or the arithmetic structure of the real line.

A set of criteria that a mathematical structure should satisfy in order to qualify as a candidate for an image processing algebra has been ascertained. From a rigorously logical point of view, the image algebra itself is mainly determined by the choice of basis. Nevertheless, it is useful to specifically articulate those properties which are desirable for the algebraic system as a whole. The image algebra must satisfy certain heuristic conditions in order to serve as the supporting structure for image processing. There are, in fact, many different bases which lead to the same algebra. Therefore, while there is interplay between the basis criteria and the algebra criteria, they are to some extent exclusive. As a result, the desirable properties for an imaging algebra need to be treated separately.

The algebra must be effective and efficient. It is effective to the extent that it enables autonomous target detection and classification algorithms to be represented and developed. Necessary to a practical effectiveness is simplicity and clarity; the algebra must be accessible to those who desire to use it. Its efficiency depends upon the extent to which it allows for algorithms to be developed in a favorable fashion with respect to cost and resources. It must allow for the ready exploitation of the parallelism which is inherent in so many imaging algorithms.

The algebra should unify many typed criteria. It should serve as a vehicle for bringing together the many diverse areas of image processing through the utilization of precise specifications.

The imaging algebra should be at once expandable and robust. Expandability requires that there will be a capability to delete, insert, or modify operators. Robustness requires that the schema should have little or no variation with changes in operators, types, or constraints. Moreover, the formalism must be adaptive to changes due to advances in mathematics, in the characteristics of imaging sensors and in the architecture of processors and memory elements.

The imaging algebra structure should support object oriented design. This requires it to be programmably transportable. It should be an easy task to go from operators in the algebra into code for most machines. The framework should also support a disciplined programming style with various levels of abstraction. It should lead to brevity, clarity, modularity, and concinnity.

Certainly all of the preceding conditions cannot be satisfied in their entirety. Nonetheless, they can serve as guidelines to which the construction of a useful and comprehensive imaging algebra might aspire.

It was recognized that any structure which encompasses the previously mentioned elemental operations must be many typed. It could be a many sorted relational structure or a many sorted partial algebra. The overall choice in this direction was a many sorted algebra. Many sorted algebras are discussed in Appendix C.

Yet the issue cannot be left at this level. Many well-known mathematical structures fit the requirements of a many-sorted algebra. Such a structure is quite general.

But the particular entity arising from a choice of basis is most specific. Indeed, it is here that the approach taken becomes crucial.

To put the matter succinctly, an image is a function whose domain is a subset of the collection of integral lattice points in the plane and whose range is the real numbers. Therefore, in looking for a basis, it is natural to examine closely the structure of the lattice points, ZxZ, and the real number line, R. Inherent operations within those mathematical structures induce, or lead to, corresponding operations on images. These so-called induced operations arise naturally and therefore must be investigated. The choice as to which ones are ultimately chosen for the imaging algebra is not a mathematical question, but a heuristic one. It requires an exhaustive examination of those operations which are utilized in image processing. Essentially the point is this: there is no need to look for exotic operations; there is only a need to take enough of the natural operations to accomplish the specification of current image processing algorithms.

The preceding approach served as a catalyst yielding the list of elemental operations.

Two questions arise at once. First, why not include more induced operations? To this one can only answer that they are not needed at the present time. Those which can be derived from the ones chosen are already there. Others, which might have been included, appear to have no place in actual image processing as it exists today. By utilizing induced operations, the basis is ipso facto expandable. If there should arise a need for more induced operators, then simply include them. Since they will be induced, they will be consistent with the already existing structure. This point is paramount for the development of an image processing language: the language can be augmented without unduly disturbing its current structure.

The other question which comes immediately to mind is this: Why not include non-induced operations? In other words, why not bring in operations which do not conform in a natural mannner to the underlying lattice structure and the underlying real number structure? Here the answer is quite straightforward. Not only have no such operations been discovered in the literature search, but it is not likely that such operations would be of much practical value. If an imaging language is to be useful as a means of communication and specification, it cannot be burdened with bizarre operations which do not conform to the usual arithmetic notions held by users. Moreover, it is not likely that bizarre operations would be utilized in algorithm development to begin with.

From a practical point of view, the imaging algebra as conceived by the contractor is a mathematical structure in its own right. Just as the term vector algebra denotes a particular algebra, which happens to be many sorted, it is likely that the term image algebra will someday come to denote a particular algebra, the one which arises naturally from the underlying mathematical structure and the pragmatics of image processing.

A description and demonstration of some of the properties and relationships defining each mathematical structure identified above has been performed. A sample of these properties can be found in Appendix C.

There were several criteria cited above concerning desirable basis properties. The proposed basis is certainly acceptable from the perspective of those criteria. It is manipulable; the basis elements provide accessible tools for macro-function development. It is sufficient; the basis provides a structured and straightforward environment for algorithm expression. It is representable; the basis has an excellent spanning capability for image to image and image to parameter operations.

In so far as the criteria described are concerned, the proposed algebra (resulting form the choice of basis) possesses simplicity and clarity. It allows, through the translation operator, excellent exploitation of parallelism. Moreover, its induced nature makes it both expandable and robust. Lastly, the elemental mathematical level of the basis operators will result in machine transportability.

The proposed basis together with necessary accompanying explanation is given in Appendix C.

An initial demonstration of the capability and versatility of the optimal mathematical structure has been completed. Appendix gives a sample of some image processing techniques which have been translated and expressed in terms of the optimal structure identified previously.

APPENDIX B

PHASE II PROPOSED ACTIVITIES FOR SPONSOR'S REVIEW AND APPROVAL

It is planned to extend the algebraic and other pertinent mathematical properties and relationships of the optimal structure. Basic research necessary to extend the mathematical properties and relationships of the selected basic set shall be conducted. This research will extend the image processing algebra's properties and relationships which have been identified in Appendix A, section 3.

In particular, an indepth investigation shall be made into determining the various equational constraints, such as the commutative or associative laws, satisfied or not satisfied by the operators in the algebra. The natural induced method by which the operators in the algebra were obtained will aid in this variety specification. This information will be useful in the determination of various substructures within the optimal structure. Well known substructures such as groups, rings, lattices and so on shall be described and identified along with their importance.

Principle operational and transformational properties and relationships of the algebra shall be identified and illustrated. This shall include operational and transformational optimization techniques.

The identification of equational constraints mentioned above can be used for transformational and operational optimization. Due to these equational constraints, various transforms, measurement techniques, or feature analysis techniques may be expressed in terms of the basis operators in several different ways. For instance, if the distributive law (of multiplication by reals with respect to image addition) holds, then there would be a choice of using one add and one multiply or one add and two multiplies.

The optimization criteria would be utilized in determining the best representations in terms of the basis. Various criteria, such a maximizing concurrent operators, minimizing certain types of operators, etc., shall be investigated.

Often sub-optimal solutions may be obtained involving macro operations. Various image processing operations are often comprised of common operators not belonging to the basis. These common operations are expressible in terms of basis operators and are termed macro operators. Macro operators could be represented in an optimal fashion in terms of basis operators as described above. In turn, image processing operations could be expressed in an optimal fashion in terms of these macro operations. It is possible that only a sub-optimal representation will be found for the image operation. In other words, one might be satisfied with optimization relative to the higher level macro operators and forego total optimization at the lowest level. In any case, an extensive investigation is planned into sub-optimal representations utilizing macro operations.

A list of theorems and proofs of all principal properties and relationships implied or required by the image processing algebra shall be prepared.

A summary of advantages and disadvantages of the chosen structure, together with a list of unsettled problems concerning the structure, shall be prepared. Imperative additional research or development shall also be recommended.

A document which reports the result of examining the feasibility of augmenting the developed image processing algebra by artificial intelligence (AI) techniques shall be prepared. If augmentation is determined feasible, included in the report will be a plan, with justifying rationale, for the integration of artificial intelligence with the developed image processing algebra.

Among the possible ways AI could be integrated into and along with the image algebra is by the development of knowledge base or expert systems. These systems could be designed to aid image processing personnel utilize the image algebra. They would represent a type of smart computer assisted instruction. Here the user might be knowledgeable in what imaging techniques to employ when, but would need the expert system to aid in utilizing the algebra or to optimize procedures within the algebra.

In a different but related direction, an expert system could be designed which incorporates various levels of the algebra, e.g., basis operation and various macros. The system will be used to aid in the development of image processing algorithms for those not knowledgeable in algorithm development.

An imaging algorithm is a finite string of imaging operations. Thus, it is composed of a longer finite string of macros, and a still longer string of basis operators. As such, a totally automated optimal algorithm development is indeed a goal. Here a brute force search could be employed combining macro or basis operators in various ways to determine algorithms. However, hueristics and subjective judgments should be employed in guiding this search, thus saving time. Developed algorithms should be played against simulated data. Subsequently, statistical packages should be employed in determining the quality of each algorithm.

In the image processing algebra developed herein, images were defined as functions from subsets of the lattice points in the plane into the reals; that is, each pixel had a real number attached to it denoting a value of grey. This value of grey is initially obtained from a sensor and the accuracy of the sensor is sometimes in question. A more realistic modeling of the situation might be to use a range of reals to denote the grey value at a given pixel. More generally, a possibility distribution might be employed to denote values of grey at pixels or at larger regions of images.

Demonstration of techniques for using the developed image processing algebra to express/approximate, analyze, and optimize the performance of existing image processing algorithms will be performed. In addition, the use of each new image processing transform technique resulting from the development of this image processing algebra shall be demonstrated.

A document entitled Standard Image Processing Algebra shall be prepared. This manuscript shall include definitions for a collection of standardized symbols which subsequently will be used to express the complete mathematical structure of the developed image processing algebra. There will also be illustrations demonstrating the application and versatility of the structure. The appropriate justification for proposing that the developed image processing algebra should serve as an Air Force standard image processing algebra shall also be given. A document and supporting explanatory vu-graphs entitled Standard Image Processing Algebra in a Nutshell, will give a synopsis of the developed standard image processing algebra. Furthermore, this document shall be suitable for publication in a technical journal.

At the completion of Phase II, the imaging algebra will be implemented in the form of an image processing language. Among the problems addressed in Phase II will be the so-called boundary value problems. These involve the manner in which the algebra expressed algorithms will handle boundary pixels and other special cases. Since the operators of the proposed basis work on subsets of ZxZ, it is felt that these boundary value problems will not require the introduction of new operations; rather, they will only require specially adapted subroutines with a given algorithm.

Another language related issue concerns interaction at the terminal level. The desire is for software that allows for an algebra-operator oriented symbolism at the keyboard. Basis level, and perhaps some macro level, operations should be accessed directly by single keys. Moreover, the characters on these keys should be indicative of the imaging operation to be performed. The thinking at this early stage is towards some sort of overlay which can be transported to various machines.

A second fundamental consideration is for AI interaction at the keyboard. The expertise gained by the developers of the algebra should be integrated into a comprehensive expert system which can impart direction and insight to users.

A time schedule of Phase II activities and milestones can be found in Appendix F.

APPENDIX C

TUTORIAL ON IMAGE ALGEBRA

1. INTRODUCTION

This report represents the culmination of approximately one year's effort to find an imaging algebra. Though much developmental work remains to be done, the skeleton of a satisfactory structure appears to have been found. It is the intent of this report to define, explain, illustrate and demonstrate the capabilities of the proposed algebraic structure.

In reading this report, several points regarding the germane attributes of the proposed structure should be kept in mind:

a. The ten underlying basis operators are elemental from the perspectives of both mathematics and machine implementation.

b. Seven of the basic operators are either range induced or domain induced, thereby rendering them both operationally and structurally familiar.

c. Two of the operators are projections, one extracting the domain of an image and the other allowing the extraction of the range. The tenth operator, the existential operator, allows the formation of an image. These three operators provide for extensive data structure manipulation and for easy movement among the sorts within the image algebra.

d. An already well-developed collection of image algorithm oriented macro-operators has been developed. Structural evaluation of these macros is in progress. The macros provide a workable vehicle for the transposition of existing image processing algorithms into the algebra.

e. The spanning capability of the proposed basis is extensive; indeed, not one strictly digital algorithm has yet been presented which is not expressable in terms of the basic ten operators.

f. By telescoping the basic operators and the macros, arranged in some sort of order of increasing complexity, it should be possible to develop an image processing language in which the user has access at all levels down to the basic set of ten. The resulting language will take full availability of the inherent parallelism within imaging algorithms.

The preceding attributes of the algebra represent, in compact form, those properties that were identified early in the project as being critical to the eventual success of the system.

2. FUNDAMENTAL OPERATORS IN THE IMAGING ALGEBRA

The most essential property of any set of fundamental operators, or basis, in an imaging algebra is its spanning capability, that is, the ability to serve as a set of elemental operations from which image processing algorithms can be constructed. Without a good spanning capacity, a basis, and hence the resulting imaging algebra, would fall short, no matter how excellent its other characteristics. The basis has the desired spanning capability while at the same time being composed of operations which are both simple and natural.

In order to appreciate the power and simplicity of the proposed basis, it is important to recognize that the construction of a satisfactory imaging algebra requires at the outset the exposure of the structures that underlie the operations specification of image processing algorithms. As with most mathematics, these primitive structures tend to be quite simple. In general, the end product of mathematical reasoning can be elaborate and difficult for the non-expert to penetrate; however, the premises from which the reasoning begins are usually not overly complex. In the case of the proposed imaging algebra, its structure must allow for the development of most current and (hopefully) future imaging transformations. These may ultimately prove to be of a high order of complexity; nevertheless, they must spring from some low level set of primitives. These, in turn, will be a by-product of the supporting mathematical structures upon which the operations are based.

Once the structural particulars of imaging algorithms are discovered, they lead directly to the proposed basis. A digital image defined herein is a partial function on ZxZ into the reals, that is, it is a function whose domain is a subset of ZxZ and whose codomain is the real number system R. An image can be described by the domain being the extent and codomain being the grey value. The set of all images will be denoted by X. It is mathematically natural to look within the structures of ZxZ and R to find the primitive operations of image processing. Both ZxZ and R are extremely rich and well-studied mathematical entities. Each has an extensive structure from which to draw. The proposed basis was developed by drawing upon those domain (ZxZ) and codomain (R) structural properties which play a role in digital image processing. As occurs through mathematics, these lead at once to corresponding properties (or, in this case, operations) within the new structure which they together induce.

Therefore, there naturally arises a set of domain induced (from ZxZ) operations and a set of codomain induced (from R) operations. In a sense, one might say that these are there to be found. For a successful image algebra, one needs to select those operations which are required for the convenient representation of digital imaging transformations.

It must be understood that while the preceding comments provide a natural approach to the basis selection problem, they do not provide a deterministic methodology. Pragmatic modeling decisions must be made. Not only does one have to search the literature to see what is going on, one must recognize that different images can have different domains within ZxZ. The decision as to how to proceed, when one desires to add two images with different domains within ZxZ, must be made in a heuristic manner. In making such decisions for the proposed basis, an attempt has been made to define the elemental operations in a way which reflects the manner in which the induced operations are most used in practice. Fortunately, it turns out that in every instance that has come to attention, other natural choices for the induced elemental operations are derivable as terms in the algebra or as macro-operators from the chosen basis set. These macro-operators are given in a later section along with a rigorous discussion of the inducement process.

There now follows a brief description of the basis elements grouped according to the manner in which they have been induced.

TABLE C-1. FUNDAMENTAL OPERATORS IN IMAGE ALGEBRA

I	Addition	$\oplus$
II	Multiplication	$\odot$
III	Maximum	$\textcircled{\vee}$
IV	Division	$\oplus$
V	Translation	T
VI	Rotation	N
VII	Reflection	D
VIII	Domain Extractor	K
IX	Parameter Extractor	G
X	Existential Operator	E

The first four fundamental operations to be introduced are range (codomain) induced, and they include addition, multiplication, maximum and division. The next three operations are translation, rotation, and reflection. They also take digital images into digital images; however, they are domain induced. The final three operations in the basis do not take images into images. They include the domain extraction operation, which takes an image and returns a subset of ZxZ, the parameter extraction operation, which maps an image into the reals, and the existential operation, which is used in creating an image.

An image is a real-valued mapping defined on a subset of the integral lattice ZxZ. Symbolically, an image is a mapping $f: A \rightarrow R$, where $A \subset Z x Z$. We also employ the customary notation for the class of all such mappings, R^A. Note that for the null set $\emptyset \subset ZxZ$, we obtain the so-called null image, $\emptyset$ has an empty domain. As for the collection of all images, we denote this class by X and

$$X = \bigcup_{A \subset ZxZ} R^A$$

Insofar as a particular grey value of an image f ∈ R^A is concerned, this is denoted by f(i,j), where (i,j) ∈ A ⊂ Z×Z. The first element of the pair, i, gives the position on the x-axis, while the second, j, gives the position on the y-axis.

Example 1: Let A = { (-1,0), (0,0), (1,0), (0,1), (1,1), (1,2) }, and define the image f ∈ R^A by

$$f(-1,0) = 2$$
$$f(\ 0,0) = 3$$
$$f(\ 1,0) = -4$$
$$f(\ 0,1) = 0$$
$$f(\ 1,1) = 1/2$$
$$f(\ 1,2) = 2$$

Graphically, one can illustrate f in the following manner:

	2			2	
	1		0	1/2	
y = 0 →	0	2	3	-4	
	-1				
		-1	0	1	2

x ↑ = 0

Each pair (i,j) corresponds to a square pixel which is given the grey value f(i,j). It should be noted that the solid lines are used to separate the coordinate values from the grey value table. Notice also that, for this example, we have used arrows to indicate the positions of the x-axis and the y-axis. In general, the actual ordered pair (i,j) is positioned in the center of the (i,j)-pixel.

One can interpret the grey value f(i,j) as the height of a three dimensional bar graph, where the darker values have greater heights. The following example illustrates this interpretation.

Example 2:

2				
1	4	2	3	1
0	3	2	1	0
	0	1	2	3

Bar graph interpretations must be made with care because images may have negative grey values due to processing.

a. Addition (Range Induced). Since each pixel in the domain of an image has a grey value which is an element of R, the real number system, and since there is a natural addition (+) in R, there is an induced addition defined as a binary operation on images. This image addition is denoted by $\oplus$ and is a basis operation. For each pixel in the intersection of the input domains, the output image has the arithmetic sum of the input grey values at that pixel. For a pixel which lies in one of the input domains but not both, the decision has been made to leave its grey value unchanged. A similar decision has been made regarding the multiplication operator and the maximum operator, each of which will be considered in turn. We define $\oplus$: $X x X \rightarrow X$ as follows:

If the domain of f is A and the domain of g is B, then the domain of $f \oplus g$ is $A \cup B$ and

$$(f \oplus g)(x,y) = \begin{cases} f(x,y) & (x,y) \in A - B \\ g(x,y) & (x,y) \in B - A \\ f(x,y) + g(x,y) & (x,y) \in A \cap B \\ \text{undefined} & (x,y) \notin A \cup B \end{cases}$$

This operation is illustrated in Figure C-1.

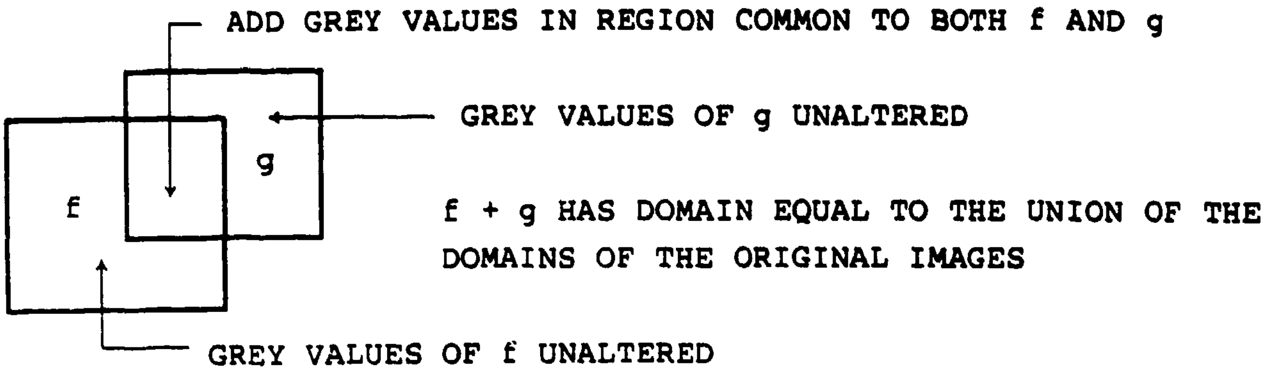

ADD GREY VALUES IN REGION COMMON TO BOTH f AND g
GREY VALUES OF g UNALTERED
g
f
f + g HAS DOMAIN EQUAL TO THE UNION OF THE DOMAINS OF THE ORIGINAL IMAGES
GREY VALUES OF f UNALTERED

Figure C-1. Venn Diagram for Addition

Example 3:

Consider the images f and g illustrated below.

f

1	2	3	1
0	4	-3	8
	0	1	2

g

2	7	8	
1	0	5	
0	1	3	
	0	1	2

The addition of f and g is $f \oplus g$ and is given in the following diagram.

$f \oplus g$

2	7	8		
1	2	8	1	
0	5	0	8	
	0	1	2	3

grey value at pixel (2,1) equals grey value of f at pixel (2,1)

grey value at pixel (0,1) is 2 + 0 = 2.

b. Multiplication (Range Induced). Similar reasoning as given in the addition operation is applied to the natural multiplication (·) in R. The result is a pixelwise induced multiplication operation on pairs of input images. For a pixel in the intersection of the domains of the input images, the corresponding grey values are multiplied. On the other hand, the grey value of a pixel which lies in only one domain of the input images is left invariant.

Hence we define the binary operator

$\odot : X \times X \rightarrow X$, where the operands are images and the output is also an image, as follows:

Let the domain of f be A and the domain of g be B. Then the domain of $f \odot g$ is $A \cup B$ and

$$(f \odot g)(x,y) = \begin{cases} f(x,y) & (x,y) \in A - B \\ g(x,y) & (x,y) \in B - A \\ f(x,y) \bullet g(x,y) & (x,y) \in A \cap B \\ \text{undefined} & (x,y) \notin A \cup B \end{cases}$$

This operation is illustrated in Figure C-2.

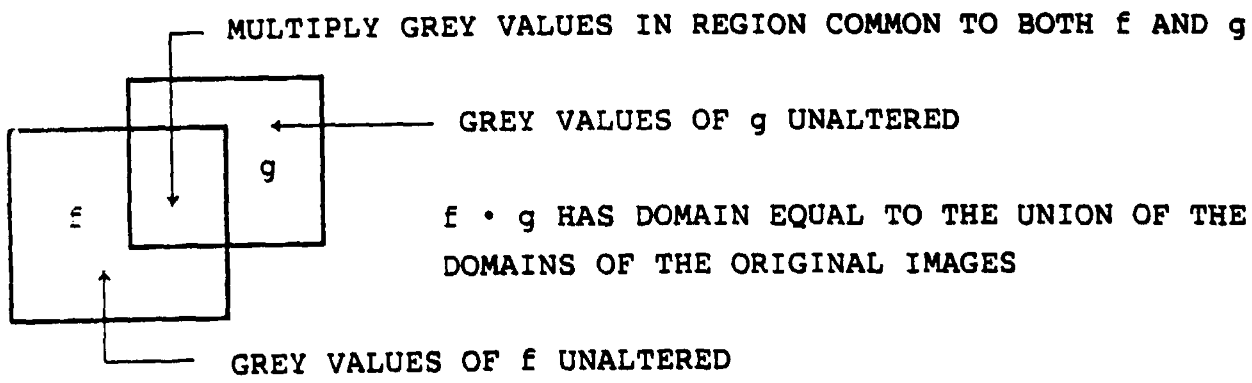

MULTIPLY GREY VALUES IN REGION COMMON TO BOTH f AND g
GREY VALUES OF g UNALTERED
g
f
f • g HAS DOMAIN EQUAL TO THE UNION OF THE DOMAINS OF THE ORIGINAL IMAGES
GREY VALUES OF f UNALTERED

Figure C-2. Venn Diagram for Multiplication

Example 4:

If f and g are given in Example 3, then $f \odot g$ is illustrated below.

2	7	8		
1	0	15	1	
0	4	-9	8	
	0	1	2	3

c. **Maximum (Range Induced).** Given two real numbers in R there is a natural order operation called maximum. Simply stated, for two real numbers y and z, $y \circledvee z$ is either y,z or their common value, depending respectively upon whether y is greater, z is greater or they are equal. This naturally induces a pixelwise maximum on the intersection of two input domains. Once again, the heuristic determination has been made to leave the input images unaltered off the intersection. The operation is denoted by $\circledvee$.

We define $\circledvee : X \times X \rightarrow X$, where the domain of f is A and the domain of g is B, then

$$(f \circledvee g) = \begin{cases} f(x,y) & (x,y) \in A - B \\ g(x,y) & (x,y) \in B - A \\ f(x,y) \vee g(x,y) & (x,y) \in A \cap B \\ \text{undefined} & (x,y) \notin A \cup B \end{cases}$$

An illustration is given in Figure C-3.

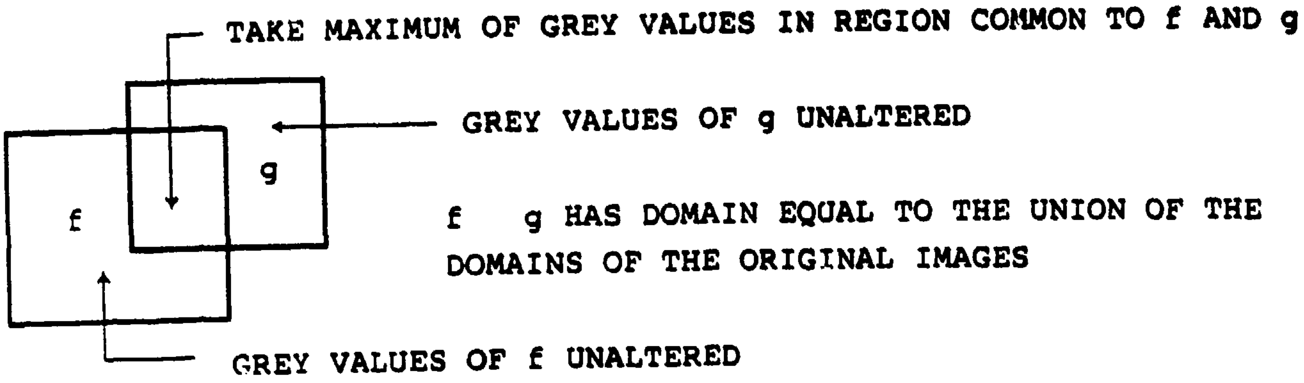

TAKE MAXIMUM OF GREY VALUES IN REGION COMMON TO f AND g
GREY VALUES OF g UNALTERED
g
f
f g HAS DOMAIN EQUAL TO THE UNION OF THE DOMAINS OF THE ORIGINAL IMAGES
GREY VALUES OF f UNALTERED

Figure C-3. Venn Diagram for Maximum

Example 5:

If f and g are given as in Example 3 then $f \circledvee g$ is illustrated below.

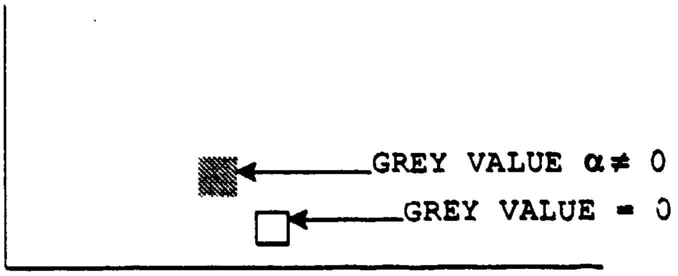
2	7	8		
1	2	5	1	
0	4	3	8	
	0	1	2	3

d. Division (Range Induced). Each grey value Z which is not zero has a reciprocal grey value 1/z. Hence there is a natural image operation, called division, which replaces each nonzero grey value by its reciprocal. It is denoted by $\oplus$. Since in R the reciprocal of zero is undefined, it has been decided that the division operation should leave the output image undefined at any pixel for which the input image has grey value zero.

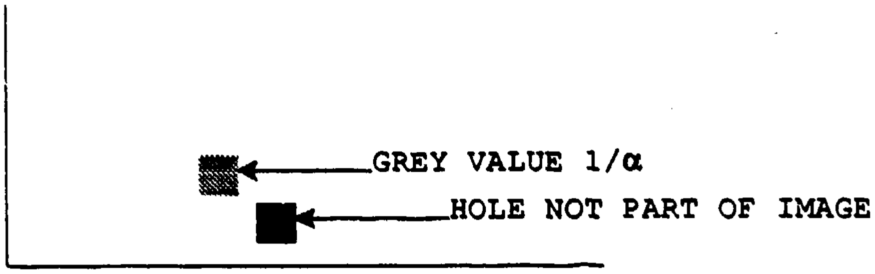
Consequently, $\oplus : X \rightarrow X$, where if there is a zero pixel in the input image then the output image has a smaller domain than the input.

Specifically, $(\oplus f)\ (x,y) = 1/f(x,y)$ if $f(x,y) \neq 0$, and is undefined if $f(x,y) = 0$. When the division is preceded by a multiplication operator $\odot$, we shall omit $\odot$. A diagram of this operation follows:

BEFORE



AFTER



Example 6:

Consider the image g given in Example 3, $\oplus g$ is given below in the illustration.

$\oplus g$

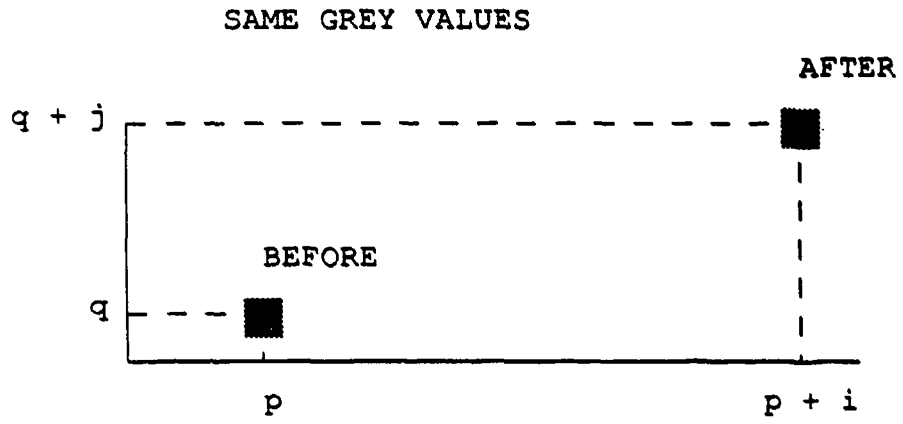
2	1/7	1/8	
1		1/5	
0	1	1/3	
	0	1	2

e. Translation (Domain Induced). Given a position vector in two dimensional space, denoted by (m,n), the vector addition between (m,n) and another vector (i,j) yields a new position vector (m+i,n+j). Geometrically, the original position is moved over (x direction) i units to the right and up (y direction) j units. This position operation induces the translation operation on images. The elemental operator T moves an image over and up, while leaving grey values unchanged. Notationally, T(f,i,j) or $f_{i,j}$ is used to indicate the image obtained by moving it over i units and up j units. It is this domain induced operator T which has proved to be invaluable in the exploitation of the natural parallelism which exists in many imaging operations. We define

T: $X \times Z \times Z \rightarrow X$, where T is the trinary operator defined by:

$$(T(f,i,j))(x,y) = f(x-i,y-j)$$

An illustration follows:



Example 7:

Consider the image f illustrated below

f

vertical location → of pixel

3				
2				
1	2	2	1	
0	4	-3	8	
	0	1	2	3

← grey value of pixel (2,1) is 1, that is f(2,1) = 1

← horizontal location of pixel

The translate of f, i=2 units to the right and j=1 units up is denoted by T(f,2,1) = $f_{2,1}$ and is illustrated below.

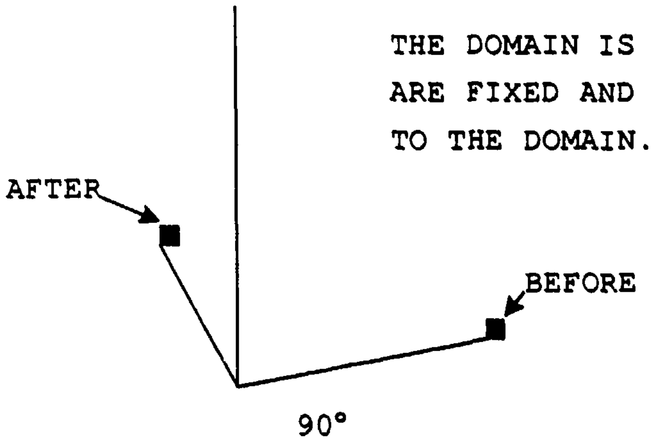
$f_{2,1}$

2			2	2	1
1			4	-3	8
0					
	0	1	2	3	4

f. Ninety Degree Rotation (Domain Induced). A set of ordered pairs in the two dimensional lattice ZxZ can be rotated 90 degrees in the counter-clockwise direction. This at once induced a ninety degree rotation operation N. The grey values of the input image are left unchanged and the image is simply rotated. Consequently: N : X → X by

$$(N(f))\ (x,y) = f(y,-x).$$

The illustration given below depicts how N rotates the image f counter-clockwise about the origin a full 90 degrees and does not otherwise alter it.



THE DOMAIN IS ROTATED, THE GREY VALUES ARE FIXED AND DO NOT CHANGE RELATIVE TO THE DOMAIN.

Example 8:

If the image g is given in Example 3, then N(g) is illustrated below.

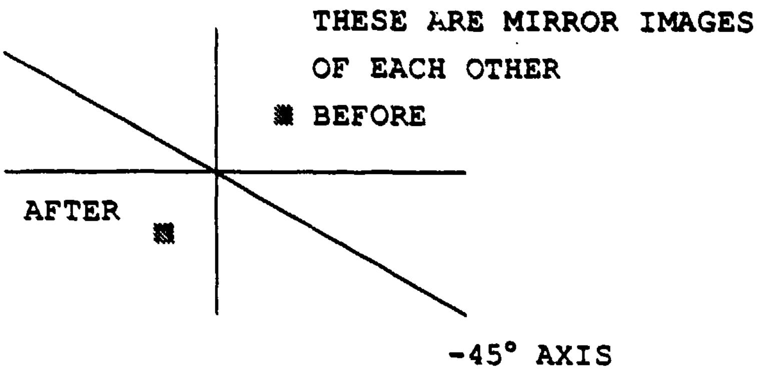
g

2	7	8	
1	0	5	
0	1	3	
	0	1	2

N(g)

2				
1	8	5	3	
0	7	0	1	
	-2	-1	0	1

g. **Diagonal Reflection (Domain Induced).** This operation is similar in origin to the 90° rotation, except that the image is flipped out of the page around a 135° line through the origin. This operation is denoted by D. D : X → X, where $(D(f))(x,y) = f(-y,-x)$. Hence D makes row pixels become column pixels (and conversely) by rotating the image 180° out of the page about the -45° axis. That is, it flips the image as illustrated below.



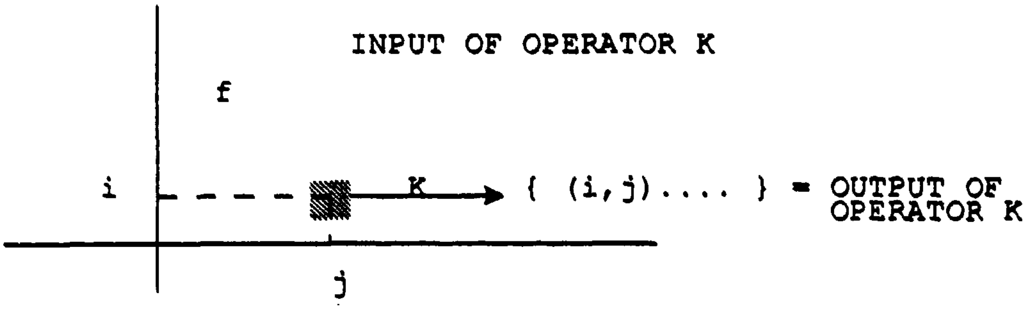
Example 9:

Consider the image g given in Example 3. Then D(g) is illustrated below.

D(g)

1				
0	7	0	1	
-1	8	5	3	
	-2	-1	0	1

h. Domain Extractor. The domain of an image is a subset of Z×Z. It is natural and convanient to consider the operation K which takes an image and yields a subset of Z×Z, that subset being the domain of the image. Hence $K : X \rightarrow 2^{Z\times Z}$ and for f in $R^A, A \subset Z\times Z$, $K(f) = A$.



Example 10:

Consider the image f illustrated in Example 3.

f

1	2	3	1
0	4	-3	8
	0	1	2

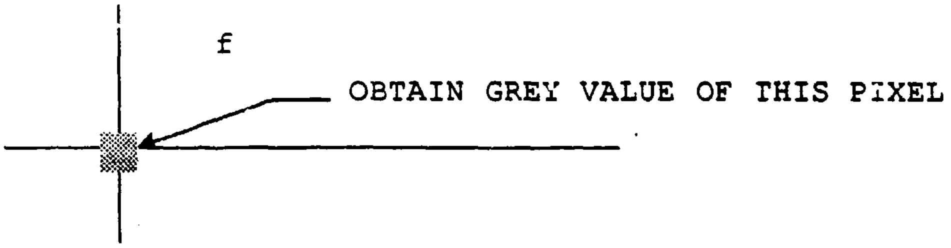
The domain extraction operation performed on f is K(f) and K(f) = { (0,0), (0,1), (1,0), (1,1), (2,0), (2,1) }.

i. Parameter Extractor. Each pixel in the domain of an image has a given grey value. It is often necessary to read out that value, which is an element of R, the codomain. It turns out that it is only necessary to assume the ability to extract the grey value at the origin pixel. Others can be found by first applying the appropriate translations. This basis operation yields the grey value at the origin pixel for a given input image. The complexity of the rigorous definition results from the desire to have this operator defined even if the grey value at the origin is undefined. In that event, the closest grey value is chosen. This latter stipulation is essentially just a mathematical formality since it is possible to move any grey value to the origin by translation. The operator G extracts the grey value of the pixel which is closest to the origin in Euclidean distance and at the smallest angle from the abscissa.

Hence G: X → R, where

$$G(f) = \begin{cases} f(0,0) & \text{when } f \in R^A \text{ and } (0,0) \in A \\ 0 & \text{when } f = \emptyset \\ f(i,j) & \text{otherwise, where } \tan^{-1}(j/i) \text{ is minimized for minimum } i^2 + j^2. \end{cases}$$

The operation is illustrated below.

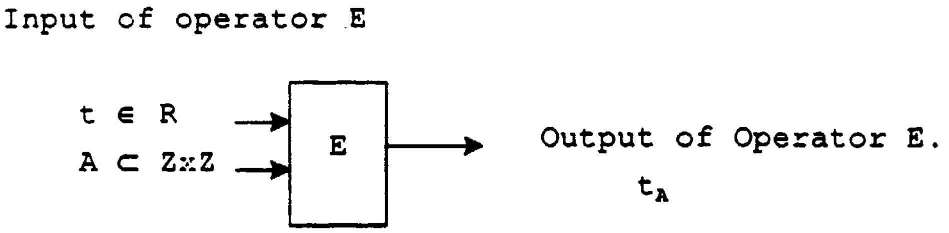


Example 11:

Consider the image f given in Example 3. Then G(f) = 4 since f(0,0) = 4.

j. **Existential Operator.** Somewhat opposite of the domain finding operation and parameter extraction operation is the Existential Operator E. This operator is a binary operator used in manufacturing an image. The inputs of this operator are a grey value t and a subset A of ZxZ. The output is a constant image with domain A such that every pixel in A has grey value t. Such an image is denoted by t_A; i.e., $t_A(i,j)=t \in R$; for $(i,j) \in A$ and is undefined elsewhere. Thus we define $E : R \times 2^{Z \times Z} \rightarrow$ where for $t \in R$ and $A \in 2^{Z \times Z}$, $E(t,A) = t_A$.

If A = ∅ then E(t,A) = ∅ (the empty image).



t_A = All pixels in A have grey value t.

Example 12:

Notice that E(2,{ (0,0), (0,1), (1,0), (1,1) }) is given by:

2			
1	2	2	
0	2	2	
	0	1	2

The aforementioned operations form the proposed basis. Each taken individually is very simple in its structure. Yet taken as a collection, they possess a powerful spanning capability insofar as the following transformation types are concerned: image to image, image to parameter and image to set. Both their simplicity and their power result from the inducement methodology (described in para 9) which brings them to light.

3. MACRO OPERATIONS IN THE IMAGING ALGEBRA

In the previous section, the fundamental operations in the imaging algebra were introduced. These operations are either directly or indirectly used in describing, representing, or expressing important image processing operations. A hierarchy of operations is a beneficial approach in this representation. The fundamental operations can be thought of as components in a system. By utilizing numerous components, subsystems are created and, as a whole, the system is comprised of several subsystems.

In the imaging algebra, macro operations act like the subsystems. Each macro operation is formed from the fundamental operations using function composition and, as such, each of the ten fundamental operations are themselves elemental macros. In the universal algebra, macro operations are also called terms for the algebra. Finally, these macro operators are ultimately employed again using function composition in forming image processing operations. These image processing operations are indirectly represented in terms of the ten fundamental operations. However, if desired, a direct representation in terms of this basis can always be given, but this expression will often be long and tedious.

In this section, several macro operations shall be described, along with their basis representations. This presentation is also given in a hierarchal manner. Once macro operations are defined, they might be employed in describing a later macro in conjunction with the utilization of fundamental operations.

In addition to defining macros which take images (that is, elements of $X = \bigcup_{A \subset Z \times Z} R^A$) into images, macros will be given involving other sorts of sets. In particular, the set of all images with finite domain will be needed. Here, $Y = \bigcup_{A \subset Z \times Z} R^A$ and it is noticed that $Y \subset X$. Additionally, the set of all binary images B will be used in defining some macro operations. In this case, $B = \bigcup_{A \subset Z \times Z} \{0,1\}^A$ and $B \subset X$. Other than these, macros will be given with domain or codomain involving the subsets of Z×Z or the reals respectively.

a. Subtraction Macro

(1) Description. Grey values in an image f can be negated. This is done by employing the subtraction operation $\ominus$ to the image f, where

$$\ominus : X \rightarrow X$$

This operator has an operand which is an image, and the result of the operation is an image with the same domain as the original image. Furthermore,

$$(\ominus f)(x,y) = -f(x,y)$$

When the unary operator $\ominus$ is preceded by $\oplus$, it is customary to omit the $\oplus$.

(2) Basis Representation. The subtraction operation is most easily found by utilizing the existential operator, along with the domain finding operator and multiplication operator. Thus,

$$\ominus\ f = E(-1,K(f)) \odot f.$$

b. Minimum Macro

(1) Description. Similar to the maximum operation previously discussed, there is the minimum operation $\owedge$. This operator, when applied to two images, gives the pixelwise minimum value of grey on the intersection of the domains of the two images. On the symmetric difference of the domains, the grey values remain unchanged. Hence

$$\owedge : X \times X \rightarrow X$$

where $\owedge$ is a binary operator which has operands that are images, and it yields an image as output. Let the domain of f be A, and the domain of g be B then

$$(f \owedge g)(x,y) = \begin{cases} f(x,y) & (x,y) \in A - B \\ g(x,y) & (x,y) \in B - A \\ f(x,y) \owedge g(x,y) & (x,y) \in A \cap B \\ \text{undefined} & (x,y) \notin A \cup B \end{cases}$$

(2) Basis Representation. The minimum macro operation is represented by employing the maximum operator and the subtraction macro:

$$f \owedge g = \ominus (\ominus f \ovee \ominus g)$$

c. **Scalar Multiplication Macro**

(1) Description. It is convenient to define a scalar multiplication operation $\triangle$ which is similar to the scalar multiplication in a vector space. Here, an image f is to be multiplied by a given real number α and the result will be an image whose grey value at a given pixel is α times the original grey value. Thus,

$$\triangle : R \; x \; X \rightarrow X$$

where $\triangle$ is a binary operation with operands being an image and a real number and the output being an image. Furthermcre,

$$(\alpha \triangle f)(x,y) = \alpha \bullet f \, (x,y)$$

This operation is also often denoted by αf.

(2) Basis Representation. The scalar multiplication macro is obtained under function composition utilizing the existential operation in addition to the domain extractor and the multiplication operations. Hence,

$$\alpha \mathbin{\triangle} f = E\ (\alpha,\ K(f)) \odot f.$$

d. Zero and One Image Macros:

(1) Description. Among all the constant images, that is images whose pixels all have the same value, the zero and one images are most useful. The zero image with domain A is denoted by 0_A.

(2) Basis Representation. Any constant image is easily found using the existential operation; indeed,

$$0_A = E(0,A)$$

and

$$1_A = E(1,A)$$

e. Complementation Macro

(1) Description. Let f be a binary image over the set A, that is, f in $\{0,1\}^A$. Then the complementary image f^c is also an element of $\{0,1\}^A$. It is defined by

$$f^c(i,j) = \begin{cases} 0 & \text{if } f(i,j) = 1 \\ 1 & \text{if } f(i,j) = 0 \\ \text{undefined} & \text{if } (i,j) \notin A \end{cases}$$

The complementation operator C is defined by

$$C : B \rightarrow B$$

by

$$C(f) = f^c$$

where

$$B = \bigcup_{A \subset Z \times Z} \{0,1\}^A$$

is the collection of all binary images.

(2) Basis Representation. Complementation is easily found using addition, subtraction and the identity image.

Let f be an element of $\{0,1\}^A$. Then

$$f^c = [\ominus f] \oplus 1_A$$

where 1_A is the image consisting of ones on the subset A of ZxZ. Notice that $1_A = E[1, K(f)]$.

f. Rotation Macros (N^2 and N^3)

(1) Description. While the basis operator N rotates 90°, it is often necessary to rotate 180° or 270°. The operators N^2 and N^3 respectively accomplish these rotations.

We define

$$N^2 : X \rightarrow X$$

by

$$N^2(f)(i,j) = f(-i,-j),$$

and

$$N^3 : X \to X$$

by

$$N^3(f)(i,j) = f(j,-i)$$

(2) Basis Representation. The above rotation operators are given by

$$N^2 = N[N(f)],$$

and

$$N^3 = N^2[N(f)]$$

g. Horizontal Reflection Macro

(1) Description. In order to reflect or flip an image around the x-axis, we define the horizontal flip macro

$$F : X \to X$$

by

$$F(f)(i,j) = f(i,-j)$$

(2) Basis Representation The operator F is given by

$$F(f) = N[D(f)]$$

where N is the 90° rotation basis operator and D is the diagonal flip basis operator.

h. Vertical Reflection Macro

(1) Description. In order to reflect an image around the y-axis we define the vertical flip macro

$$V : X \to X$$

by

$$V(f)(i,j) = f(-i,j)$$

(2) Basis Representation. The vertical flip macro is given by

$$V(f) = N^3[D(f)]$$

where N^3 is the 270° rotation basis operator and D is the diagonal flip basis operator.

i. 45° Reflection Macro

(1) Description. In order to reflect an image around the 45° line $y = x$ we define the 45° diagonal reflection operator

$$D_0 : X \rightarrow X$$

by

$$D_0(f)(i,j) = f(j,i) \quad .$$

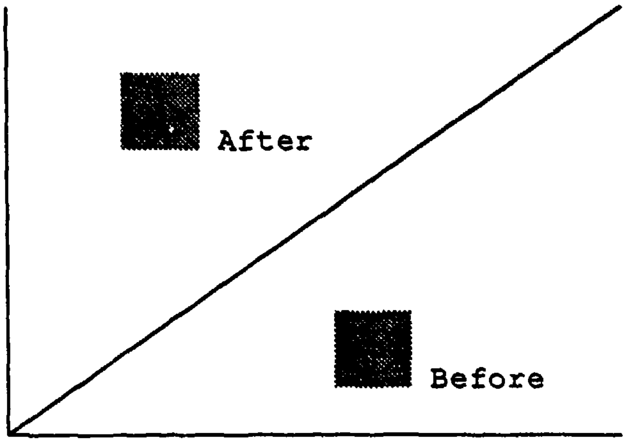


Figure C-4. Illustration of 45° Reflection Macro

(2) Basis Representation. The 45° diagonal reflection

macro is given by

$$D_0(f) = N[F(f)],$$

where N is the 90° rotation basis operator and F is the horizontal flip macro.

j. Zero Divide Macro

(1) Description. At times, it is useful to consider a division operation which takes images into images by taking the reciprocal of non-zero grey values at a given pixel and leaving zero grey values unchanged. Consequently, we define

$$\oplus_0 : X \rightarrow X$$

by

$$\oplus_0 f(i,j) = \begin{cases} 1/f(i,j) & \text{if } f(i,j) \neq 0 \\ 0 & \text{if } f(i,j) = 0 \\ \text{undefined} & \text{if } (i,j) \notin K(f) \end{cases}$$

When $\oplus$ is preceded by $\odot$, we often leave out the multiplication symbol $\odot$.

(2) Basis Representation. This operation is found by using division, addition, and the zero constant image.

Let f be an element of R^{λ}. Then

$$\oplus_0 f = (\oplus f) \oplus 0_{\lambda}$$

where $0_{\lambda} = E[0,K(f)]$.

k. Threshold Macro

(1) Description. Numerous threshold operations exist; however, they all are binary operations which take an image and a real number into a binary image. We begin with one variation of threshold. The threshold operator is defined as follows:

$$\tau : X \times R \rightarrow B$$

by

$$\tau(f,t)(i,j) = \begin{cases} 1 & \text{if } f(i,j) \geq t \\ 0 & \text{if } f(i,j) < t \\ \text{undefined} & \text{if } (i,j) \notin K(f) \end{cases}$$

To simplify notation, we usually write

$$\tau\,(\,\cdot\,) = \tau(\,\cdot\,,t)$$

In particular, we shall most often concern ourselves with thresholding at 0, and in this case τ_0 represents the operation at hand.

(2) Basis Representation. This operation is constructed from the minimum, zero divide, complement, subtraction, scalar multiplication, as well as zero and one images. Let f be an element of R^A and t be a real number.

Then

$$\tau_0(f) = [(f \owedge 0_A) \oplus_0 (f \owedge 0_A)]^c$$

and

$$\tau_t(f) = \tau_0(f \ominus t \triangle 1_A)$$

1. Variations of Thresholding Macros

(1) Description. The threshold operator τ_t has been defined utilizing the inequality

$$f(i,j) \geq t$$

Four variations of the underlying threshold operation will be defined in accordance with the solution sets of the following equations:

$$\begin{array}{ll} (a) & f(i,j) \leq t \\ (b) & f(i,j) > t \\ (c) & f(i,j) < t \\ (d) & f(i,j) = t \end{array}$$

The corresponding macro threshold operators will be respectively denoted by τ^1 , τ^2, τ^3, and τ^4. They are respectively defined by:

$$(a) \quad \tau^1_t(f)(i,j) = \begin{cases} 1 & f(i,j) \leq t \\ 0 & f(i,j) > t \end{cases}$$

$$(b) \quad \tau^2_t(f)(i,j) = \begin{cases} 1 & f(i,j) > t \\ 0 & f(i,j) \leq t \end{cases}$$

$$(c) \quad \tau^3_t(f)(i,j) = \begin{cases} 1 & f(i,j) < t \\ 0 & f(i,j) \geq t \end{cases}$$

$$(d) \quad \tau^4_t(f)(i,j) = \begin{cases} 1 & f(i,j) = t \\ 0 & f(i,j) \neq t \end{cases}$$

(2) Basis Representation. Let f and t be elements of R^A and R respectively. Then

$$\begin{array}{ll} (a) & \tau^1_t(f) = \tau_{-t}(\ominus f) \\ (b) & \tau^2_t(f) = [\tau^1_t(f)]^c \\ (c) & \tau^3_t(f) = [\tau_t(f)]^c \\ (d) & \tau^4_t(f) = \tau_t(f) \owedge \tau^1_t(f) \end{array}$$

m. Clipper Macro

(1) Description. The threshold operation gives a binary image which has grey value 1 on those pixels in which f exceeds or is equal to some given input threshold value. The clipper acts in an analogous fashion in that it leaves f unaltered on the pixels for which it is greater than or equal to some threshold value and it sets the image equal to zero where it is less than that threshold value. We define

$$CL : X \times R \rightarrow X$$

by

$$CL(f,t)(i,j) = \begin{cases} f(i,j) & \text{if } f(i,j) \geq t \\ 0 & \text{if } f(i,j) < t \\ \text{undefined} & \text{if } (i,j) \notin K(f) \end{cases}$$

(2) Basis Representation. The clipper operation is found using the thresholding macro in addition to multiplication. Let f be an image and t a real number. Then

$$CL(f,t) = f \odot \tau_t(f)$$

Notice that corresponding to the four variations of thresholding, there are four variations of clipping. These are

$$CL^k(f,t) = f \odot \tau^k{}_t(f)$$

for k = 1,2,3 and 4.

n. Positive and Negative Part Macros

(1) Description. The positive and negative part macro operations take images into images, and both yield images without negative grey values. The positive part of an image f is the

image

$$f^{+}(i,j) = \begin{cases} f(i,j) & \text{if } f(i,j) \geq 0 \\ 0 & \text{if } f(i,j) < 0 \end{cases}$$

The negative part of an image f is the image

$$f^{-}(i,j) = \begin{cases} -f(i,j) & \text{if } f(i,j) \leq 0 \\ 0 & \text{if } f(i,j) > 0 \end{cases}$$

(2) Basis Representation. The positive part of an image is found using thresholding and multiplication, while the negative part uses subtraction in addition to these other operations. Thus,

$$f^{+} = \tau_0(f) \odot f$$

and

$$f^{-} = [\tau_0(\ominus f)] \odot [\ominus f]$$

It is interesting to notice that

$$f^{+} = CL(f,0)$$

and

$$f = f^{+} \ominus f^{-}$$

o. **Absolute Value Macro**

(1) Description. Given an image f, the absolute value operator yields an image whose pixel values are the absolute values of the original pixel values. Therefore,

$$AB : X \rightarrow X$$

where the usual notation for absolute value is often used instead of the prefix notation. Thus,

$$AB(f) = | f |$$

In any case, $| f |$ is defined as

$$| f |(i,j) = \begin{cases} f(i,j) & \text{if } f(i,j) \geq 0 \\ -f(i,j) & \text{if } f(i,j) < 0 \\ \text{undefined} & \text{if } (i,j) \notin K(f) \end{cases}$$

(2) Basis Representation Absolute value is defined in terms of the positive and negative parts. For any image f, we have

$$AB(f) = f^{+} \oplus f^{-}$$

p. **Support Macro**

(1) Description. The support of an image is often defined to be the subset of pixels where the image is defined and has grey value not equal to 0. In a somewhat similar manner, the support macro, supp, is defined to be an unary operator which takes an image into a binary image:

$$supp : X \rightarrow B$$

where

$$supp(f)(i,j) = \begin{cases} 1 \text{ if } f(i,j) \neq 0 \\ 0 \text{ if } f(i,j) = 0 \end{cases}$$

(2) Basis Representation. The support macro is defined in terms of the thresholding macro, the complementation macro, the absolute macro, and the subtraction operation:

$$supp(f) = (\tau_0[\ominus |f|])^c$$

q. Addition Macro

(1) Description. It is often convenient to have an addition operator which adds two images only on the intersection of the domains and has that intersection as the domain of the final output. We define

$$\mathit{a} : X \times X \to X$$

by

$$\mathit{a}(f,g)(i,j) = \begin{cases} f(i,j) + g(i,j) & \text{for } (i,j) \in A \cap B \\ \text{undefined} & \text{elsewhere} \end{cases}$$

where f has domain A and g has domain B in Z×Z.

(2) Basis Representation. The macro is defined using function composition involving the fundamental addition, division, and multiplication along with the zero and one constant image. Let f be an element of R^A and g be an element of R^B. Then

$$\mathit{a}(f,g) = \oplus [\oplus ([f \oplus g] \odot [1_{A\cap B} \oplus 0_{A\cup B}])] \oplus 0_{A\cap B}$$

r. Multiplication Macro:

(1) Description. It is often convenient to have a multiplication operator which multiplies two images only on the intersection of the domains and has that intersection as the domain of the final output. We define

$$\mathcal{M} : X \times X \to X$$

by

$$\mathcal{M}(f,g)(i,j) = \begin{cases} f(i,j) \bullet g(i,j) & \text{for } (i,j) \in A \cap B \\ \text{undefined} & \text{elsewhere} \end{cases}$$

where f has domain A and g has domain B in $Z \times Z$.

(2) Basis Representation. Let f be an element of R^A and g be an element of R^B. Then

$$\mathcal{M}(f,g) = \mathcal{A}(f \odot g,\ 0_{A \cap B})$$

s. Divide Macro

(1) Description. Whenever we write $f \oplus g$, by convention we mean $f \odot [\oplus g]$. As a result, if the domains of f and g are A and B respectively then the output image is defined on the intersection of $\{(i,j) \in B$ such that $g(i,j) \neq 0\}$ with A. One might wish the output domain to be a subset of the domain of f. We define

$$\boldsymbol{D} : X \times X \rightarrow X$$

by

$$\boldsymbol{D}(f,g)(i,j) = \begin{cases} f(i,j)\ /\ g(i,j) & \text{if both f and g are defined at (i,j) and if } g(i,j) \neq 0 \\ \text{undefined} & \text{elsewhere} \end{cases}$$

(2) Basis Representation. This macro is described in terms of the multiplication macro and the fundamental division operation:

$$\boldsymbol{D}(f,g) = \mathcal{M}(f \oplus\ g)$$

t. Higher (Maximum Macro)

(1) Description Like the addition macro and the multiplication macro, there exists a maximum macro, called higher. It is given by

$$\mathcal{H}: X \times X \rightarrow X$$

where

$$\mathcal{H}(f,g)(i,j) = \begin{cases} f(i,j) \vee g(i,j) & \text{for } (i,j) \in A \cap B \\ \text{undefined} & \text{elsewhere} \end{cases}$$

A being the domain of f and B being the domain of g.

(2) Basis Representation. Let f be an element of R^A and g be an element of R^B. Then

$$\mathcal{H}(f,g) = \mathcal{M}[f \ominus g, 1_{A \cap B}]$$

Note that

$$1_{A \cap B} = \mathcal{M}[1_A, 1_B]$$

u. Lower (Minimum Macro)

(1) Description. The macro $\mathcal{L}$ is defined in a manner analogous to the macro $\mathcal{H}$ except that the minimum is involved. Indeed,

$$\mathcal{L}: X \times X \rightarrow X$$

by

$$\mathcal{L}(f,g)(i,j) = \begin{cases} f(i,j) \wedge g(i,j) & \text{for } (i,j) \in A \cap B \\ \text{undefined} & \text{elsewhere} \end{cases}$$

where A is the domain of f and B is the domain of g.

(2) Basis Representation. Let f be an element of R^A and g be element of R^B. Then

$$\mathcal{L}(f,g) = \mathcal{M}[f \owedge g, 1_{A \cap B}].$$

v. **Special Zero Macro Operators**

(1) Description. The following two macros are related to the support macro. They are the zero indicator macro

$$\mathcal{Z} : X \rightarrow B$$

by

$$\mathcal{Z}(f)(i,j) = \begin{cases} 1 \text{ if } f\ (i,j) = 0 \\ \text{undefined elsewhere} \end{cases}$$

and the zero retainer macro

$$\mathcal{Z}' : X \rightarrow B$$

by

$$\mathcal{Z}'(f)(i,j) = \begin{cases} 0 \text{ if } f\ (i,j) = 0 \\ \text{undefined elsewhere} \end{cases}$$

(2) Basis Representation. The zero indicator macro operations are representable in terms of the support macro, the complement macro and the division operation. The zero retainer is obtained from the zero indicator using the multiplication macro and the zero identity image. The zero indicator is given by

$$\mathcal{Z}(f) = \oplus[\ \text{supp}(f)\ ]^c$$

and the zero retainer is given by

$$\mathcal{Z}'(f) = \mathcal{M}[0_A, \mathcal{Z}(f)]$$

where A is domain of f.

w. Selection Operator

(1) Description. The selection macro $\mathcal{S}$ is similar to operations on data bases. In the imaging algebra, $\mathcal{S}$ takes an image f in R^A and a subset B of $Z \times Z$ and returns an image g which has domain $A \cap B$ and is equal to f on that domain. Hence

$$\mathcal{S} : X \times 2^{Z \times Z} \rightarrow X$$

where

$$\mathcal{S}(f,B)(i,j) = \begin{cases} f(i,j) & \text{for } (i,j) \in A \cap B \\ \text{undefined elsewhere} & \end{cases}$$

and A is the domain of f.

(2) Basis Representation For f in R^A and subset B of $Z \times Z$, $\mathcal{S} = \mathcal{M}(f, 1_B)$ where $\mathcal{M}$ is the multiplication macro and 1_B the image with grey values equal to 1 on B.

x. Extension Macro

(1) Description. The extension macro takes an image f (the primary image) and an image g (the secondary image), and outputs a new image. This image is identical to f on the domain of f and is identical to g on that part of the domain of g which is outside the domain of f. Hence,

$$\mathcal{E} : X \times X \rightarrow X$$

by

$$\mathcal{E}(f,g)(i,j) = \begin{cases} f(i,j) & (i,j) \in A \\ g(i,j) & (i,j) \in B - A \\ \text{undefined} & (i,j) \notin A \cup B \end{cases}$$

where f is in R^A, g is in R^B and B - A is the subtraction of B from A.

(2) Basis Representation The extension operation is found using the selection macro in addition to using scalar multiplication, multiplication and addition. With f in R^A,

$$\mathcal{E}(f,g) = [(0 \triangle \mathcal{S}(g,A)) \odot g] \oplus f$$

y. **Grey Level Summation**

(1) Description. Given an image f in Y (the set of all images with finite domain) the grey level summation macro Σ_0 outputs a real number which is the sum of the grey levels in f. Hence define

$$\Sigma_0 : Y \rightarrow R$$

by

$$\Sigma_0(f) = \sum_{(i,j)\in A} f(i,j)$$

where f has domain A.

(2) Basis Representation. This operation is described in terms of the selection operation, the translation operation, addition operation and the grey value parameter extractor. Let f be an element of R^A. Then

$$\Sigma_0(f) = G[\sum_{(i,j)\in A} \mathcal{S}[T_{-i,-j}(f), \{(0,0)\}]]$$

Where G is the grey value functional: G gives the grey value at the origin and the summation implies a repeated use of the fundamental addition operation $\oplus$. It is interesting to note that, by employing N^2, $\Sigma_0(f)$ can be written as

$$\Sigma_0(f) = G[\sum_{(i,j)\in B} \mathcal{S}[T_{i,j}(f), \{(0,0)\}]]$$

where B is the domain of $N^2(f)$.

z. Grey Level Product

(1) Description. Given an image f with finite domain, the grey level product macro outputs a real number which is the product of the grey levels in f. We define

$$\Pi_0 : Y \rightarrow R$$

where

$$\Pi_0(f) = \prod_{(i,j)\in A} f(i,j)$$

and f has domain A.

(2) Basis Representation. Let f be an element of R^A with $A \subset Z \times Z$ and card $A < \infty$. Then

$$\Pi_0(f) = G\left[\prod_{(i,j)\in A} \mathcal{S}\left[T_{-i,-j}(f), \{ (0,0) \} \right] \right]$$

In the basis representation, $\prod_{(i,j)\in A}$ denotes a repeated use of the fundamental multiplication operation $\odot$ and G gives the grey value at (0,0).

aa. Grey Level Maximum

(1) Description. Given an image f with finite domain A, the grey level maximum macro outputs a real number which is the maximum of the grey level in f. We define

$$\textcircled{\vee}_0 : Y \rightarrow R$$

by

$$\textcircled{\vee}_0 (f) = \bigvee_{(i,j)\in A} f(i,j)$$

(2) Basis Representation. Let f be an element of R^A where $A \subset Z \times Z$ and card $A < \infty$. Then

$$\textcircled{$\vee$}_0 (f) = G [\bigvee_{(i,j)\in A} \mathcal{S} [T_{-i,-j}(f), \{ (0,0) \}]]$$

In the basis representation $\bigvee_{(i,j)\in A}$ denotes a repeated use of the fundamental maximum operation $\textcircled{$\vee$}$ and G the grey value extractor.

bb. Grey Level Minimum

(1) Description. Given an image f with bounded domain, the grey level minimum macro outputs a real number which is the minimum of the grey levels in f. Hence

$$\textcircled{$\wedge$}_0 : Y \rightarrow R$$

where

$$\textcircled{$\wedge$}_0(f) : \bigwedge_{(i,j)\in A} f(i,j)$$

and f has domain A.

(2) Basis Representation. Let f be an element of R^A, where $A \subset Z \times Z$ and card $A < \infty$. Then

$$\textcircled{$\wedge$}_0(f) = G [\bigwedge_{(i,j)\in A} \mathcal{S} [T_{-i,-j}, (f), \{ (0,0) \}]$$

In the basis representation $\bigwedge_{(i,j)\in A}$ denotes the repeated use of the minimum macro operation $\textcircled{$\wedge$}$ and G is the grey value extractor.

cc. Restriction Macro

(1) Description. The extension operator extends a given image into the domain of a secondary image. The new domain is the union of the original domains. The restriction macro defined

herein restricts the primary image to the intersection of its domain with the domain of the secondary image. Notice that no new grey values are defined by this operator. Thus

$$\mathfrak{R} : X \times X \rightarrow X$$

by

$$\mathfrak{R}\ (f,g)(i,j) = \begin{cases} f(i,j) & \text{for } (i,j) \in A \cap B \\ \text{undefined elsewhere} & \end{cases}$$

where A is the domain of f and B is the domain of g.

(2) Basis Representation. The macro is obtained from the selection macro and the domain finding macro. Let f and g be elements of R^A and R^B respectively. Then

$$\mathfrak{R}(f,g) = \boldsymbol{S}(f,B) = \boldsymbol{S}(f,K(g)).$$

dd. Dot Product

Description. Suppose two images f and g have the same (finite) domain, say A. Then a dot product can be formed between f and g according to the definition.

$$\boldsymbol{D}_0(f,g) = \sum_{(i,j)\in A} f(i,j) \times g(i,j)$$

If the images do not have the same domain, then the dot product is undefined. As a result, $\boldsymbol{D}_0$ is not defined on $X \times X$. Instead

$$\boldsymbol{D}_0 : \bigcup_{A\subseteq Z\times Z,\ \text{card}\ A < \infty} R^A \times R^A \rightarrow R$$

(2) Basis Representation Let f and g be elements of R^A. Then

$$\boldsymbol{D}_0(f,g) = \Sigma_0[f \odot g]$$

ee. Filtering Macro

(1) Description. One of the most common operations in image processing is that of filtering an image by a given mask. Since a mask is nothing but an image which is being used for a specific purpose, the filtering macro needs to make no reference to the term mask. We define

$$\mathfrak{I} : Y \times Y \rightarrow Y$$

by

$$\mathfrak{I}(f,g)(i,j) = \begin{cases} \sum_{(u,v)\in B} f(i+u, j+v) \bullet g(u,v) & \text{if all terms in the sum are defined, and B is the domain of g.} \\ \text{undefined} & \text{if there exists at least one undefined term in the sum.} \end{cases}$$

(2) Basis Representation. Let f and g be elements of R^A and R^B respectively. Then

$$\mathfrak{I}(f,g) = \sum_{(i,j)\in A} E\,[\boldsymbol{\mathcal{D}}_0[\mathfrak{R}(f, T_{i,j}(g)),\ T_{i,j}(g)],\ \{(i,j)\}]$$

where $\sum_{(i,j)\in A}$ denotes the repeated use of $\oplus$ addition.

ff. Pixelwise Norms for Image Vectors

(1) Description. At times, consideration must be given to a vector of images of the form $(f_1, f_2, \ldots, f_m)$ where each f_k is an image. For the present, this discussion will be restricted to the case where all the f_k have the same domain. Once such a vector of images exists, a norming image can be defined, i.e., an image which has at each pixel the grey value which results from applying some given norm to the vector of grey values which correspond to that given pixel. In other words, for each pixel (i,j) in the common domain of the f_k, a real valued vector can be associated.

$$V_{i,j} = (f_1(i,j),\ f_2(i,j),\ \ldots,\ f_m(i,j))$$

Any norm can then be applied to $V_{i,j}$; however, this discussion is restricted to the following

$$||V_{i,j}||_\infty = \max_{k = 1,2,\ldots,m} \{|f_k(i,j)|\}$$

$$||V_{i,j}||_1 = |f_1(i,j)| + |f_2(i,j)| + |f_3(i,j)| + \ldots + |f_m(i,j)|$$

$$||V_{i,j}||_2 = (\sum_{k=1}^{m}[f_k(i,j)]^2)^{\frac{1}{2}}$$

Three corresponding operators are defined:

N_∞ : $X \times X \times \ldots \times X \rightarrow X$ (m terms in product)
N_1 : $X \times X \times \ldots \times X \rightarrow X$ (m terms in product)
N_2 : $X \times X \times \ldots \times X \rightarrow X$ (m terms in product)

These are respectively defined by

$$N_\infty(f_1,\ f_2,\ \ldots,\ f_m)(i,j) = ||V_{i,j}||_\infty$$
$$N_1(f_1,\ f_2,\ \ldots,\ f_m)\ (i,j) = ||V_{i,j}||_1$$
$$N_2(f_1,\ f_2,\ \ldots,\ f_m)\ (i,j) = ||V_{i,j}||_2$$

(2) Basis Representation. The respective basis representations of the preceding three operators are:

$$N_\infty(f_1,\ f_2,\ \ldots,\ f_m) = \bigvee_{k=1}^{m}|f_k|$$

$$N_1(f_1,\ f_2,\ \ldots,\ f_m) = \sum_{k=1}^{m}|f_k|$$

$$N_2(f_1,\ f_2,\ \ldots,\ f_m) = \{\sum_{k=1}^{m}[|f_k|]^2\}^{\frac{1}{2}}$$

where in the last representation, the notation $g^{\frac{1}{2}}$, g an image means to obtain a new image by taking the positive square root of each pixel, assuming, of course, that g has no negative grey values. It should be noted that the square root operator might be expressed in terms of basis operators, in which case, it would be only a finite approximation to the actual square root. For instance, a few terms of the Newton Raphson may be employed. In any case some convention must be adopted regarding the square root when implementing it as a procedure.

gg. Gradient Type Edge Detector

(1) Description. Many edge detection techniques involve filtering by two directional masks, one which detects change in the horizontal direction and one which detects change in the vertical direction. Examples are the usual gradient, the Prewitt gradient and the Sobel gradient. These operators each have three popular variants, the particular variant depending upon the choice of norm. Consequently, we will introduce three edge detection macro operators, one for each norm. Define

$$E_\infty : X \times X \times X \times R \rightarrow B$$
$$E_1 : X \times X \times X \times R \rightarrow B$$
$$E_2 : X \times X \times X \times R \rightarrow B$$

by

$$E_\infty(f,M,N,t) = \tau_t[N_\infty\{\mathfrak{I}(f,M),\mathfrak{I}(f,N)\}]$$
$$E_1(f,M,N,t) = \tau_t[N_1\{\mathfrak{I}(f,M),\mathfrak{I}(f,N)\}]$$
$$E_2(f,M,N,t) = \tau_t[N_2\{\mathfrak{I}(f,M),\mathfrak{I}(f,N)\}]$$

In each of the above, M and N are images which, in practice, represent directional masks.

(1) Basis Representation. The above definitions are themselves basis representations.

A particular instance of the gradient macro occurs with the Sobel edge detector. In this case, if the image is f, then the input masks are

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

As images, elements of X, these are given by

$$M = \begin{array}{r|c|c|c|} \hline +1 & -1 & 0 & 1 \\ \hline 0 & -2 & 0 & 2 \\ \hline -1 & -1 & 0 & 1 \\ \hline & -1 & 0 & +1 \end{array}$$

and

$$N = \begin{array}{r|c|c|c|} \hline +1 & 1 & 2 & 1 \\ \hline 0 & 0 & 0 & 0 \\ \hline -1 & -1 & -2 & -1 \\ \hline & -1 & 0 & +1 \end{array}$$

The threshold parameter t depends upon some heuristic or knowledge-based a priori decision.

hh. Compass Gradient Operators

Description. In a manner similar to the gradient type operators, we shall describe a compass gradient type macro operator. It can be used with any set of eight compass gradient masks such as the Kirsh or 3-level masks. We shall leave out the thresholding operation. It can always be composed with the output of the present macro to produce a binary edge image.

$$E_C : X \times X \times \ldots \times X \rightarrow X \text{ (8 terms in product)}$$

by

$$E_C(f, M_1, M_2, \ldots, M_8) = N_\infty[\mathfrak{I}(f, M_1), \mathfrak{I}(f, M_2), \ldots, \mathfrak{I}(f, M_8)]$$

where in practice M_1, $M_2, \ldots,$ M_8 are masks of the same dimension.

An example of the compass gradient macro occurs when $M_1, M_2, \ldots,$ M_8 are the eight Kirsch masks. Technically, each Kirsh mask is an image in R^A, where A consists of the origin together with its immediate neighbors, both strong and weak. For instance,

$$M1 = \begin{array}{r|r|r|r|}
\hline
+1 & -5 & 3 & 3 \\ \hline
0 & -5 & 0 & 3 \\ \hline
-1 & -5 & 3 & 3 \\ \hline
 & -1 & 0 & 1
\end{array}$$

ii. Binary Morphological Macro Operators

(1) Description. Because of their prevalence in image processing, it is useful to define macro operators for the four basic morphological operations. Each is of the form

$$\text{Operator: } (B \cap Y) \times (B \cap Y) \rightarrow (B \cap Y)$$

The four macros are dilation, erosion, opening and closing. They are respectively given by

$$DI(f,g) = \bigvee_{(i,j)\in A} T_{i,j}(g) \quad \text{(A is the domain of f)}$$

$$\mathrm{ER}(\;,g) = [\mathrm{DI}\{f^{c},N^{2}(g)\}]^{c}$$

$$\mathrm{OP}(f,g) = \mathrm{ER}[\mathrm{DI}\{f,N^{2}(g)\},N^{2}(g)]$$

$$\mathrm{CL}(f,g) = \mathrm{DI}[\mathrm{ER}(f,g),g]$$

(2) Basis Representation. The preceding expressions are basis representations.

jj. Grey Level Morphological Macro Operators

(1) Description. The following four grey level counterparts to the binary morphological macros given in paragraph ii above are in use. It should be noted that there are other ways to generalize the binary morphological operators. The following operators are of the form

$$\text{Operator:}\quad Y \times Y \rightarrow Y$$

They are grey level dilation, grey level erosion, grey level opening, and grey level closing. The are respectively defined by

$$\hat{D}\hat{I}(f,g)](i,j) = E[\mathbin{\bigcirc\hspace{-0.75em}\vee}_{0}\{\boldsymbol{a}(f,g_{i,j})\},\ (i,j)\ ]$$

$$\hat{E}\hat{R}(f,g)](i,j) = E[\mathbin{\bigcirc\hspace{-0.75em}\vee}_{0}\{\boldsymbol{a}(f,\ominus g_{i,j})\},\ (i,j)\ ]$$

$$\hat{O}\hat{P}(f,g) = \hat{E}\hat{R}[\hat{D}\hat{I}\{f,N^{2}(g)\},N^{2}(g)]$$

$$\hat{C}\hat{L}(f,g) = \hat{D}\hat{I}[\hat{E}\hat{R}(f,g),g]$$

Several comments are in order:

(a) It is assumed that the structuring element g is symmetric.

(b) In words, $\hat{D}\hat{I}$ (f,g) is evaluated at (i,j) by adding the (i,j) translation of g to that part of the image f which matches domainwise the translation of g, taking the maximum value of that addition and then making that maximum the grey value at (i,j).

(c) $\hat{E}\hat{R}$ is similar to $\hat{D}\hat{I}$ except that $g_{i,j}$ is subtracted and a minimum is taken.

(d) Given $\hat{D}\hat{I}$ and $\hat{E}\hat{R}$, $\hat{O}\hat{P}$ and $\hat{C}\hat{L}$ are evaluated in a manner similar to the manner in which $\hat{D}\hat{I}$ and $\hat{E}\hat{R}$ are obtained from $\hat{O}\hat{P}$ and $\hat{C}\hat{L}$.

(2) Basis Representation. The preceding expressions are basis representations.

4. ANALYTIC MACRO OPERATORS

Consider the complex-valued function b(z) of a complex variable. Suppose that b(z) is analytic in the disk of radius r centered at the origin. Then b(z) can be expressed as a power series:

$$b(z) = \sum_{k=0}^{\infty} b_k z^k$$

If b(z,n) denotes the nth partial sum of the series, then the meaning of the series expression is that $\lim_{n\to\infty} b(z,n) = b(z)$ for all z such that $|z| < r$. The purpose of this section is to formalize an extension of analytic functions to the construction of particular macro-operators in the image algebra.

Suppose b(z) is analytic and possesses the preceding power series representation. Moreover, let f be an image and n be any non-negative integer. If f is the null image, then we define b(f,n) to also be the null image. If f is not null, then we define b(f,n) by

$$b(f,n) = \sum_{k=0}^{n-1} b_k f^k$$

where

a. for n > 0, $f^n = \prod_{k=0}^{n-1} f$

the product being the image product, $\odot$, of f with itself n times.

b. $f^0 = E[1,K(f)]$.

c. $b_k f$ is the scalar multiplication of b_k times f in the image algebra.

d. The summation denotes image summation, $\oplus$.

When viewed as a binary operator with multi-typed inputs f and n, b is an operator in the image algebra. It is called an analytic operator. It should be noted that the definition of b(f,n) is a basis representation in and of itself.

Now suppose pixel (i,j) is in the domain of image f. By construction, b(f,n) is domain stable and hence (i,j) is in the domain of b(f,n). Moreover, due to the pixelwise definitions of the induced basis operators $\odot$, $\oplus$, and $\triangle$,

$$[b(f,n)]\ (i,j) = \sum_{k=0}^{n-1} b_k f(i,j)^k$$

In other words, the grey value of b(f,n) at the (i,j) pixel is the n^{th} partial sum of $b(f(i,j))$.

$$[b(f,n)]\ (i,j) = b[f(i,j),n].$$

Although we could have easily defined limits in the image algebra by inducing them from the range, we have not done so since our purpose has been to remain in the digital mode. Nonetheless, since $[b(f,n)]\ (i,j)$ is a real number for each n, it does make sense to take the limit of the sequence $\{[b(f,n)]\ (i,j)\}$ as $n \to \infty$. Since it is assumed that $b(z)$ is analytic in the disk of radius r, if $|f(i,j)| < r$, then

$$\lim_{n\to\infty} [b(f,n)]\ (i,j) = \lim_{n\to\infty} b[f(i,j),n] = b[f(i,j)]$$

Put simply, if all grey values of f have absolute value between -r and r, then each pixelwise limit is equal to the value of the original analytic function at the grey value of the input image. Moreover, the result remains unchanged if $f(i,j) = u(i,j) + iv(i,j)$ is a complex-valued image and the operations in the image algebra are applied by means of complexification.

We now present some particularly important instances of analytic macro-operators. In all cases, the fundamental point to keep in mind is that each is expressable directly in terms of the range induced basis elements.

EXP: The exponential function e^z is analytic in the entire complex plane. It has the power series representation

$$e^z = \sum_{k=0}^{\infty} \frac{z^k}{k!}$$

The corresponding analytic macro-operator in the image algebra is called the exponential operator and is defined by

$$\mathrm{EXP}(f,n) = \sum_{k=0}^{n} \frac{f^k}{k!}$$

$$= E[1,K(f)] \oplus f \oplus \frac{1}{2!}[f \odot f] \oplus \ldots \oplus \frac{1}{n!}[\ f \odot f \odot \ldots \odot f]$$

where the last basis product in the expression involves n multiplicands. Note that each grey value of EXP(f,n) is a finite series approximation to the exponential of the corresponding grey value of the input image f; indeed,

$$[\mathrm{EXP}(f,n)](i,j) = \sum_{k=0}^{n} \frac{f(i,j)^k}{k!}$$

Moreover, since e^z is analytic in the entire complex plane, for each (i,j) in the domain of f, regardless of the grey value,

$$\lim_{n \to \infty} [\mathrm{EXP}(f,n)](i,j) = e^{f(i,j)}$$

(Note that this limit is taken outside of the image algebra)

Example: Let f be the image defined by f(0,0) = 2, f(0,1) = 0, f(1,0) = -1, and f(-1,0) = 1:

2				
1		0		
0	1	2	-1	
	-1	0	1	2

Then EXP(f,2) is given by:

$$[\mathrm{EXP}(f,2)](-1,0) = 1 + 1 + \frac{1}{2} = \frac{5}{2}$$

$$[\mathrm{EXP}(f,2)](0,0) = 1 + 2 + \frac{2^2}{2} = 5$$

$$[\mathrm{EXP}(f,2)](1,0) = 1 + (-1) + \frac{(-1)^2}{2} = \frac{1}{2}$$

$$[\mathrm{EXP}(f,2)](0,1) = 1 + 0 + 0 = 1$$

Hence, the output image looks like

2				
1		1		
0	$\frac{5}{2}$	5	$\frac{1}{2}$	
	-1	0	1	2

Now suppose we consider EXP(f,n) for arbitrary n. If pixelwise limits are taken then

$$\lim_{n\to\infty} [\text{EXP}(f,n)](-1,0) = e$$

$$\lim_{n\to\infty} [\text{EXP}(f,n)](0,0) = e^2$$

$$\lim_{n\to\infty} [\text{EXP}(f,n)](1,0) = e^{-1}$$

$$\lim_{n\to\infty} [\text{EXP}(f,n)](0,1) = 1$$

As noted previously, these limits take place outside the image algebra.

COS: Like e^z, the cosine function, cos(z), is analytical in the entire plane, with

$$\text{COS}(z) = \sum_{k=0}^{\infty} (-1)^k \frac{z^{2k}}{(2k!)}$$

Consequently, the associated macro-operator COS is defined in the following manner: If n is even, then

$$\text{COS}(f,n) = \sum_{k=0}^{n/2} (-1)^k \frac{f^{2k}}{(2k!)}$$

$$= E[1,K(f)] \ominus \frac{1}{2!} [f \odot f] \oplus \ldots \oplus (-1)^{n/2}\frac{1}{n!}[f \odot f \odot \ldots \odot f]$$

If n is odd, then

$$\mathrm{COS}(f,n) = \sum_{k=0}^{(n-1)/2} (-1)^k \frac{f^{2k}}{(2k)!}$$

The reason for the complicated definition is that, in terms of the power series representation for cosine, all odd terms are missing. In order to maintain consistency, COS(f,n) must be defined by using n + 1 terms (including the 0 term) of the series. This means that, for n even, the actual series representation of COS(f,n) will have n/2 + 1 terms.

As usual, each grey value of COS(f,n) is a finite series approximation to the cosine of the corresponding grey value of f. Since cos(z) is analytic in the entire plane, for each (i,j) in the domain of f,

$$\lim_{n\to\infty} [\mathrm{COS}(f,n)](i,j) = \cos[f(i,j)]$$

Example: Consider the image f defined by $f(0,0) = \pi/2$, $f(1,0) = 0$, and $f(0,1) = 1$. Then the image COS(f,5) is defined by

$$[\mathrm{COS}(f,5)](0,0) = 1 - (\pi/2)^2/2 + (\pi/2)^4/24$$
$$[\mathrm{COS}(f,5)](1,0) = 1 - 0 + 0 = 1$$
$$[\mathrm{COS}(f,5)](0,1) = 1 - 1/2 + 1/24 = 13/24$$

As we did for the exponential macro, consider COS(f,n) for arbitrary n. Then

$$\lim_{n\to\infty} (\mathrm{COS}(f,n))(0,0) = 0$$
$$\lim_{n\to\infty} (\mathrm{COS}(f,n))(1,0) = 1$$
$$\lim_{n\to\infty} (\mathrm{COS}(f,n))(0,1) = \cos(1)$$

SIN: The situation for the SIN macro is exactly analogous to that holding for COS; however, in this instance, the even terms of the power series for sin(z) are missing. As a result, we define, for even n > 0,

$$\mathrm{SIN}(f,n) = \sum_{k=0}^{n/2-1} (-1)^k \frac{f^{2k+1}}{(2k+1)!}$$

For n odd,

$$\mathrm{SIN}(f,n) = \sum_{k=0}^{(n+1)/2-1} (-1)^k \frac{f^{2k+1}}{(2k+1)!}$$

For any (i,j) in the domain of f,

$$\lim_{n\to\infty} [\mathrm{SIN}(f,n)](i,j) = \sin[f(i,j)]$$

It is important to pay attention to the relation between the analytic macros EXP, COS, and SIN, and the functions exponential, cosine and sine functions which exist in whatever machine is being employed for digital image processing. Let us denote these machine functions by EXPONENTIAL, COSINE, and SINE, respectively. Each of these machine functions is an approximation. For the moment, suppose each is computed by a truncated power series expansion with a fixed number of terms, say N. Then for any image f and for any (i,j) in the domain of f,

$$[\mathrm{EXP}(f,N)](i,j) = \mathrm{EXPONENTIAL}[f(i,j)],$$
$$[\mathrm{COS}(f,N)](i,j) = \mathrm{COSINE}[f(i,j)],$$
$$[\mathrm{SIN}(f,N)](i,j) = \mathrm{SINE}[f(i,j)].$$

In other words, the image algebra macro-operators produce, in a pixelwise manner, the corresponding machine functions. As a result, for a given machine, one can define machine macros accordingly. These would be written EXPONENTIAL(f), COSINE(f), and SINE(f), and they would be defined by the preceding equations. If such an approach is taken, one salient point must be kept in mind:

$$[\mathrm{EXPONENTIAL}(f)](i,j) \neq e^{f(i,j)}$$

since the machine function is a power series truncation defined by the macro operator EXP(f,N). Moreover, similar comments apply to COSINE and SINE.

Consequently, there is no macro operator $e^{f(i,j)}$ in the algebra, nor can there be. Nor can there be a function e^x in the machine. As a result, nothing is gained by employing the symbology EXPONENTIAL except for a certain simplicity when employing the macro EXP(f,N). Moreover, the use of such a notion would make the algebra machine-dependent, and machine dependency would render the entire structure less effective. Therefore, we shall not specify macros such as EXPONENTIAL.

Before leaving this section, an important point concerning analytic macros should be made.

Suppose f is an image for which $|f(i,j)| < 1$ for all (i,j) in the domain of f. Consider the operator H defined by

$$H(f) = \oplus [E\{1,K(f)\} \ominus f].$$

For any point (i,j) in the domain of f,

$$[H(f)](i,j) = [1 - f(i,j)]^{-1}$$

Now consider the function $H(z) = \frac{1}{(1-z)}$, which is analytic in the open disk of radius 1 in the complex plane. This function has the power series representation

$$H(z) = \sum_{k=0}^{\infty} z^k$$

Hence, it induces the analytic macro-operator

$$H_0(f,n) = \sum_{k=0}^{n} f^k$$

Of particular interest is the relationship between H and H_0.

For any (i,j) in the domain of image f,

$$\lim_{n\to\infty} [H_0(f,n)](i,j) = \sum_{k=0}^{\infty} f(i,j)^k$$

$$= [1 - f(i,j)]^{-1}$$

$$= [H(f)](i,j)$$

In a pixelwise sense, $\lim_{n\to\infty} H_0(f,n) = H(f)$. But this limit relationship exists outside the algebra! The macro H depends only upon the fact that R is a field; however, the pixelwise limit involves the topology of R.

More can be said regarding the limit relationship. Since $H(z) = \frac{1}{(1-z)}$ holds for $|z| < 1$, the limit is only good for $|f(i,j)| < 1$. However, both H and H_0 are defined for all images. Once again, the algebraic character of the image algebra makes itself felt. H_0 and H depend only upon the manner in which the field properties of the range have been induced into the algebra. In fact, relative to the algebra itself, the notion of an analytic macro is purely fictitious. Indeed, any image sum of the form

$$\sum_{k=0}^{n} a_k f^k$$

makes complete sense. The relevance of the analyticity of the inducing function b(z) manifests itself only outside the algebra. Nevertheless, the formality of the notion does not render it superfluous. On the contrary, the situation is analogous to the relationship between the rational numbers, which can be expressed without reference to the Dedekind cut axiom (or some equivalent axiom), and the complete space of real numbers.

Note, moreover, that even going outside the image algebra, the notion of analyticity is only relevant to those images whose grey values remain within the disk of convergence.

5. MATRIX TYPE MACRO-OPERATORS

In practice, one often works with images which possess rectangular domains. An image which will be called rectangular if its domain is of the form I x J, where I consists of a set of n consecutive integers, say $\{x, x+1, \ldots, x+n-1\}$ and J consists of a set of m consecutive integers, say $\{y, y+1, \ldots, y+m-1\}$. In order to normalize matrix operations in the image algebra, we shall call a rectangular image a matrix image if $x = 0$ and $y = -m + 1$. The image pixels can be viewed as m by n matrix, though it must be remembered that it is still an image. A matrix image has the form

$$\begin{pmatrix} f(0,0) & f(1,0) & \cdots\cdots & f(n-1,0) \\ f(0,-1) & f(1,-1) & \cdots\cdots & f(n-1,-1) \\ \vdots & \vdots & & \vdots \\ f(0,-m+1) & f(1,-m+1) & \cdots\cdot & f(n-1,-m+1) \end{pmatrix}$$

The intention here is to embed the collection of m by n matrices into the collection X of images in a canonical fashion. Using this canonical injection, it will be shown that the standard operations on matrices can be accomplished within the image algebra. In order to simplify notation, we shall let F<m,n> denote the collection of all m by n matrix images.

To begin with, it will be shown that matrix multiplication can be represented in terms of the basis. To facilitate matters, we shall let R^n_j denote the row of n grid points given by

$$R^n_j = \{(0,j), (1,j), \ldots, (n-1,j)\}$$

Moreover, C_i^m denotes the column of m grid points given by

$$C_i^m = \{(i,0), (i,-1), \ldots, (i,-m+1)\}$$

If f is an element of F<m,n> and g is an element of F<n,r>, then we define the matrix multiplication of f by g in the usual manner. If h denotes the matrix product, then h is an element of F<m,r>, and for (i,j) in the canonical m by r rectangular domain,

$$h(i,j) = \sum_{k=0}^{n-1} f(k,j)g(i,-k)$$

One should take care to note that the i determines the column and the j determines the row.

The methodology to achieve a basis representation is quite straightforward since the matrix product at (i,j), h(i,j), is nothing but a dot product of the j^{th} row of f with the i^{th} column of g. To accomplish this end, we select out the j^{th} row of f, that selection being $\mathcal{S}(f,R_j^n)$. We also select out the i^{th} column of g, that selection being $\mathcal{S}(g,C_i^n)$. The latter selection is rotated 90°and then translated so that it has the same domain as $\mathcal{S}(f,R_j^n)$. The dot product of the resulting image,

$$T[N\{\mathcal{S}(g,C_i^n)\},0,-(i-j)]$$

is then taken with $\mathcal{S}(f,R_j^n)$. This dot product gives the value h(i,j). The existential operator is then employed to produce an image with a singleton domain, {(i,j)} and grey value h(i,j). Tne outputs of the above procedure are then added (by basis addition) to produce the desired m by r product image. Written out, the basis representation of matrix multiplication is given by

$$\sum_{i=0}^{n-1} \sum_{j=0}^{-m+1} E[\mathcal{D}_0[\mathcal{S}(f,R_j^n),T\{N[\mathcal{S}(g,C_i^n)],0,-(i-j)\}], \{(i,j)\}]$$

Henceforth, we shall denote the matrix multiplication macro by f*g.

Under basis addition, scalar multiplication and matrix multiplication, the collection F<n,n> of square matrix images is a linear algebra.

Before proceeding, we note that, for any matrix f in F<m,n>, the transpose is given by D(f), the diagonal flip of f. Moreover, we define the identity of dimension n, I_n, by

$$I_n(i,j) = \begin{cases} 1 \text{ if } j = -i \\ 0 \text{ otherwise} \end{cases}$$

for each (i,j) in the rectangular grid defining the canonical domain for F<n,n>.

We now wish to demonstrate that determinants of square matrix images can be found through the use of basis operations. For a 1 by 1, the problem is trivial. Suppose f is 2 by 2. Then the determinant of f, det(f), is given by

$$\begin{aligned} \det(f) &= f(0,0)f(1,-1) - f(1,0)f(0,-1) \\ &= G[\boldsymbol{S}(f,\{(0,0)\})_{0,0} \odot \boldsymbol{S}(f,\{(1,-1)\})_{-1,1}] \\ &= -G[\boldsymbol{S}(f,\{(1,0)\})_{-1,0} \odot \boldsymbol{S}(f,\{(0,-1)\}_{0,1})] \end{aligned}$$

where G is the parameter extractor. In general, for an n by n matrix, the determinant is given numerically by

$$\det(f) = \sum_{p} s_p \, f(0,-p(0)) \odot f(1,-p(1)) \;\; . \; . \; . \;\; \odot f((n-1),-p(n-1))$$

where:

a. The summation is taken relative to real number addition.

b. The summation is taken over all permutations p(k) of the n integers 0, 1, 2, ..., n-1.

c. s_p is either +1 or -1, depending on whether the permutation p is respectively even or odd.

In a direct generalization of the case for a 2 by 2 matrix, the determinant in the n by n case has the basis representation

$$G[\ \sum_{p} s_p \prod_{k=0}^{n-1} \boldsymbol{S}(f,\{(k,-p(k))\})_{-k,p(k)}\ ]$$

where:

a. The summation is relative to basis addition.

b. The product is relative to basis multiplication.

c. The subscript denotes translation in the x-direction by -k, and in the y-direction by p(k).

It should be noted that the macro-operators for matrix multiplication and the determinant are somewhat peculiar in that it appears that they require special types of input images. In the first case, for f*g, the number of columns of f must equal the number of rows g; moreover, each must be situated properly in the grid. In fact, the situation is no different than for the dot product macro $\boldsymbol{D}_0$, where the two input images must share a common domain.

Regarding the determinant, the problem is a bit more complicated. Not only must f be an element of F<n,n>, but there is, in reality, a different determinant macro for each class F<n,n>, since the definition involves the dimension n.

Nevertheless, we shall continue to act as though there were a single determinant, det(f), and we will not indulge in such nuances as the writing of $\det_n(f)$.

Our next goal is to achieve a basis representation for the matrix multiplicative inverse of a matrix image. We shall employ the well-known adjoint method. Consequently, we must develop representations of the cofactors. The cofactor operator, COF, requires three inputs, a matrix image f and two integers. COF(f,i,j) is $(-1)^{i+j}$ times the determinant obtained by deleting the j^{th} row and the i^{th} column. (Once again, note the interchanged roles of i and j.) For convenience, we define four sets of pixels:

a. $B^1_{i,j} = \{0, 1, \ldots, i-1\} \times \{0, -1, \ldots, -j+1\}$

b. $B^2_{i,j} = \{i+1, i+2, \ldots, n-1\} \times \{0, -1, \ldots, -j+1\}$

c. $B^3_{i,j} = \{0, 1, \ldots, i-1\} \times \{-j-1, -j-2, \ldots, -m+1\}$

d. $B^4_{i,j} = \{i+1, i+2, \ldots, n-1\} \times \{-j-1, -j-2, \ldots, -m+1\}$

A basis representation for the cofactor macro is given by

$$\begin{aligned} \mathrm{COF}(f,i,j) = (-1)^{i+j} \det [\mathcal{S}(f,B^1_{i,j}) &\oplus \mathcal{S}(f,B^2_{i,j})_{-1,0} \\ &\oplus \mathcal{S}(f,B^3_{i,j})_{0,1} \\ &\oplus \mathcal{S}(f,B^4_{i,j})_{-1,1}] \end{aligned}$$

Once the cofactors have been constructed, it is easy to define the adjoint matrix of an image in F<n,n>. Indeed, for f in F<n,n>, the adjoint, ADJ(f), is given by the pixelwise relation

$$[\mathrm{ADJ}(f)](i,j) = \mathrm{COF}(f,j,i)$$

The grey value of the adjoint image at pixel (i,j), the pixel at the intersection of the i^{th} column and j^{th} row, is given by the cofactor which is obtained by deleting the j^{th} column and the i^{th} row. Recalling that, for an element of F<n,n>, the diagonal flip gives the transpose, a basis representation for the adjoint is given by

$$\mathrm{ADJ}(f) = D[\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} E(\mathrm{COF}(f,i,j),\{(i,-j)\})]$$

where the summations are taken relative to basis addition.

The adjoint method for finding the inverse of a matrix can now be employed. Assuming f to be nonsingular, det(f) is nonzero, then the multiplicative matrix inverse of f is given by

$$f^{-1} = \frac{1}{\det(f)} \mathrm{ADJ}(f)$$

Direct substitution yields a basis representation.

Perhaps the best way to illustrate the foregoing inverse methodology is to work out an example in detail. Let f be the following image:

0	1	0	1
-1	2	2	1
-2	0	1	1
	0	1	2

f is an element of F<3,3>.

We first find det(f) by directly employing the basis representation. There are six permutations of (0, 1, 2). These are:

$$
\begin{aligned}
p_1 &= (0,\ 1,\ 2), \text{ with } s_1 = +1\\
p_2 &= (0,\ 2,\ 1), \text{ with } s_2 = -1\\
p_3 &= (1,\ 0,\ 2), \text{ with } s_3 = -1\\
p_4 &= (1,\ 2,\ 0), \text{ with } s_4 = +1\\
p_5 &= (2,\ 0,\ 1), \text{ with } s_5 = +1\\
p_6 &= (2,\ 1,\ 0), \text{ with } s_6 = -1
\end{aligned}
$$

There are six summands in the basis representation. The first is given by

$$
\begin{aligned}
&s_1[\mathcal{S}(f,\{(0,0)\})_{0,0} \odot \mathcal{S}(f,\{1,-1\})_{-1,1} \odot \mathcal{S}(f,\{(2,-2)\})_{-2,2}]\\
&\qquad = \langle 1\rangle \odot \langle 2\rangle \odot \langle 1\rangle = \langle 2\rangle
\end{aligned}
$$

where < > denotes an image having singleton domain {(0,0)} and single grey value. The second summand is given by

$$
\begin{aligned}
&s_2\ [\mathcal{S}(f,\{(0,0)\})_{0,0} \odot \mathcal{S}(f,\{1,-2\})_{-1,2} \odot \mathcal{S}(f,\{(2,-1)\})_{-2,1}]\\
&\qquad = (-1)\quad [\langle 1\rangle \odot \langle 1\rangle \odot \langle 1\rangle] \qquad \langle -1\rangle
\end{aligned}
$$

The third summand is given by

$$
\begin{aligned}
&s_3\ [\mathcal{S}(f,\{(0,-1)\})_{0,1} \odot \mathcal{S}(f,\{1,0\})_{-1,0} \odot \mathcal{S}(f,\{(2,-2)\})_{-2,2}]\\
&\qquad = (-1)\quad [\langle 2\rangle \odot \langle 0\rangle \odot \langle 1\rangle] = \langle 0\rangle
\end{aligned}
$$

The fourth summand is given by

$$\langle 2\rangle \odot \langle 1\rangle \odot \langle 1\rangle = \langle 2\rangle$$

The fifth is <0> and the sixth is also <0>. Therefore, the basis sum over p_1 through p_6 yields <3>. Lastly,

$$\det[f] = G[<3>] = 3$$

We must now obtain the cofactors. We employ the following sets in the grid:

$$B^1_{0,0} = \emptyset, \quad B^2_{0,0} = \emptyset, \; B^3_{0,0} = \emptyset$$
$$B^4_{0,0} = \{(1,-1), \quad (2,-1), \quad (1,-2), \quad (2,-2)\}$$
$$B^1_{1,0} = \emptyset, \quad B^2_{1,0} = \emptyset, \; B^3_{1,0} = \{(0,-1),(0,-2)\}$$
$$B^4_{1,0} = \{(2,-1), \; (2,-2)\}$$
$$B^1_{2,0} = \emptyset, \quad B^2_{2,0} = \emptyset,$$
$$B^3_{2,0} = \{(0,-1), \quad (0,-2), \quad (1,-1), \quad (1,-2), \; \},$$
$$B^4_{2,0} = \emptyset,$$
$$B^1_{0,-1} = \emptyset, \quad B^2_{0,-1} = \{(1,0),(2,0)\},$$
$$B^3_{0,-1} = \emptyset, \quad B^4_{0,-1} = \{(1,-2),(2,-2)\},$$
$$B^1_{1,-1} = \{(0,0)\}, \; B^2_{1,-1} = \{(2,0)\}$$
$$B^3_{1,-1} = \{(0,-2)\}, \; B^4_{1,-1} = \{(2,-2)\}$$

The remaining four collections of,

$$B^k_{2,-1} \quad , \quad B^k_{0,-2} \; , \; B^k_{1,-2} \; , \text{ and } B^k_{2,-2}$$

can be found similarly.

To find the nine cofactors, we directly employ the basis representation:

$$\mathrm{COF}(f,0,0) = (-1)^{0+0} \det[\emptyset \oplus \emptyset \oplus \emptyset \oplus \mathcal{S}(f, B^4_{0,0})_{-1,1}] = 1$$

since $\mathcal{S}(f, B^4_{0,0})_{-1,1}$ is given by

0	2	1
-1	0	1
	0	1

$$\mathrm{COF}(f,1,0) = (-1)^{1+0}\det[\, \varnothing \oplus \varnothing \oplus \mathcal{S}(f, B^{3}_{1,0})_{0,1} \oplus \mathcal{S}(f, B^{4}_{1,0})_{-1,1}] = (-1)(2) = -2,$$

since $\mathcal{S}(f, B^{3}_{1,0})_{0,1} \oplus \mathcal{S}(f, B^{4}_{1,0})_{-1,1}$ is given by

0	2	1
-1	1	1
	0	1

The remaining cofactors can be similarly computed:

$$
\begin{aligned}
\mathrm{COF}(f,2,0) &= 2\\
\mathrm{COF}(f,0,-1) &= 1\\
\mathrm{COF}(f,1,-1) &= 1\\
\mathrm{COF}(f,2,-1) &= -1\\
\mathrm{COF}(f,0,-2) &= -2\\
\mathrm{COF}(f,1,-2) &= 1\\
\mathrm{COF}(f,2,-2) &= 2
\end{aligned}
$$

Applying the existential operator and basis summing over the indices gives the cofactor image:

0	1	-2	2
-1	1	1	-1
-2	-2	1	2
	0	1	2

The adjoint is found by transposing (taking the diagonal flip). ADJ(f) is given by:

$$\begin{array}{r|c|c|c|}
\hline
0 & \frac{1}{3} & \frac{1}{3} & \frac{-2}{3} \\
\hline
-1 & \frac{-2}{3} & \frac{1}{3} & \frac{1}{3} \\
\hline
-2 & \frac{2}{3} & \frac{-1}{3} & \frac{2}{3} \\
\hline
 & 0 & 1 & 2
\end{array}$$

A direct calculation shows that $f*f^{-1} = I_3$.

6. DISCRETE PICTURE TRANSFORMS IN THE IMAGE ALGEBRA

If f is a rectangular image, then the discrete picture transform involves a pre- and post-matrix multiplication of f. In order to make this separate multiplication meaningful in the context of the image algebra, some stipulations must be made.

In the image algebra, a rectangular image is one which has a rectangular domain. However, matrix multiplication is only defined for matrix images, those which are elements of F<m,n>, for some m, n > 0. Prior to any matrix multiplication, the rectangular image must be translated so that it is a matrix image. Subsequent to pre- and post-multiplication, it can be translated back to its appropriate position. The original translation must be T[f,-i(f),-j(f)], where i(f) is the minimal value of i for which f is defined and j(f) is the maximal value of j for which f is defined.

Any discrete picture transform requires two regular matrices of a given form. If the input rectangular matrix is of dimensions m by n, then the pre-multiplication matrix must be m by m, while the post-multiplication matrix must be n by n. Moreover, in order to employ matrix multiplication within the image algebra, it is necessary to require that the pre-multiplication matrix, P, and the post-multiplication matrix, Q, are elements of F<m,m> and F<n,n>, respectively. This requirement is merely formal, since the linear algebras F<m,m> and F<n,n> are ismorphic to the corresponding linear algebras of regular matrices.

Given the preceding stipulations, a discrete picture transform on the space of m by n rectangular matrices is of the form

$$\Psi(f) = \{P*f_{-i(f),-j(f)}*Q\}_{+i(f),+j(f)}$$

As mentioned previously, the subscripts denote translations to and from F<m,n>. They play no role whatsoever in the actual transform process. Consequently, we will employ the customary discrete picture transform methodology:

$$\Psi(f) = P*f*Q$$

No generality is lost, since any procedure can always begin with a translation and end with an inverse translation.

Note that the representation P*f*Q is easily corrected to a basis representation, since * (matrix multiplication) possesses a basis representation. Consequently, those image processing operations which are of the discrete picture transform type are within the scope of the image algebra.

If it happens that the pre-multiplication matrix P and the post- multiplication matrix Q are nonsingular, then the discrete picture transform is invertible; indeed,

$$f = P^{-1}*\Psi(f)*Q^{-1}$$

where, of course, P^{-1} and Q^{-1} denote the matrix multiplicative inverses of P and Q, respectively, within the image algebra. In other words, for nonsingular P and Q, $\Psi(f)$ is invertible within the image algebra

Some examples of the discrete picture transform will now be presented.

a. Discrete Fourier Transform

The discrete Fourier transform (DFT) results from setting $P = F_{mm}$ and $Q = F_{nn}$, where F_{pp} is the p by p image matrix with grey values

$$F_{pp}(k,j) = \frac{1}{p}\, e^{i[2\pi/p]kj}$$

at pixel (k,j), where i denotes the imaginary square root of -1 and $0 \leq k \leq p-1$, $-p+1 \leq j \leq 0$. Note that we are assuming the complex values for pixels in the image algebra. Specifically,

$$F_{pp}(k,j) = \frac{1}{p}\cos[\frac{2\pi}{p}kj] + i\frac{1}{p}\sin[\frac{2\pi}{p}kj]$$

Also note that $F_{pp}(k,j)$ is not given by the exponential to a negative power. This is because the row number j is already negative in the grid enumeration scheme. Now, for f in F<m,n>, the DFT is given by

$$\Psi(f) = F_{mm}*f*F_{nn}$$

Since F_{pp} is nonsingular, inversion is given by

$$f(f) = F_{mm}^{-1}*\Psi(f)*F_{nn}^{-1}$$

b. Hadamard Transform

The Hadamard transform results from the discrete picture transform by employing the Hadamard matrices Hjj, where Hjj is situated in the space F<j,j>. Since Hjj is nonsingular, we have the transform pair

$$\Psi = H_{mm} * f * H_{nn}$$

and

$$f = H_{mm}^{-1} * \Psi(f) * H_{nn}^{-1} = \frac{1}{mn} H_{mm} * \Psi(f) * H_{nn}$$

Other commonly employed instances of the discrete picture tranform are the Haar transform, the slant transform and the discrete cosine transform. All of these are of the form, P*f*Q, with P and Q nonsingular.

Prior to leaving this section, several comments are in order regarding transforms such as the DFT and the discrete cosine transform. In the DFT, it is necessary to employ the matrix whose terms are given by

$$C_{pp}(k,j) = \frac{1}{p} \cos[\frac{2\pi}{p} kj]$$

Assuming the irrational number π to be given by some fixed rational approximation, there still exists the computation problem relative to the cosine. Of course, we could assume that the function COSINE exists within the machine; however, as demonstrated earlier, this function can be treated as a particular case of the analytic macro-operator COS(f,N) for some N. In particular, for any value x,

$$\cos(x) = G[COS(<x>,N)],$$

where

(1) G is the parameter extractor.

(2) $<x>$ denotes the image with singleton domain $\{(0,0)\}$ and single grey value x.

(3) N is a fixed integer value which corresponds to whatever power series approximation is being employed to compute the value of the cosine.

Similar comments apply to sin(x) and e^x, x real:

$$\sin(x) = G[COS(<x>,N)]$$

$$e^x = G[EXP(<x>,N)].$$

As a result of the foregoing considerations, the Fourier cosine matrix C_{pp} can be generated within the image algebra starting with the matrix image whose grey values are given by

$$z_{pp}(k,j) = kj,$$

for $k = 0, 1, \ldots, p - 1$ and $j = 0, 1, \ldots, -p + 1$.

The matrix image C_{pp} is given by

$$C_{pp} = \frac{1}{p} \cos[\frac{2\pi}{p} z_{pp}, N]$$

A similar relation holds for the Fourier sine matrix. Moreover, if appropriate complexification technicalities are taken into account, the Fourier matrix can be written as

$$F_{pp} = \frac{1}{p} EXP[i\frac{2\pi}{p} z_{pp}, N]$$

One can even go to a lower level of the algebra and consider image matrix construction within the algebra. By this we mean that, given the integer p, a matrix of the form z_{pp} could be produced by using image algebra operations. Indeed,

$$z_{pp} = \sum_{k=0}^{p-1} \sum_{j=0}^{-p+1} E[kj,\{(k,j)\}]$$

In other words, the DFT $F_{mm}*f*F_{nn}$ can be looked upon as a unary macro-operator within the image algebra. We can say this, since the dimensions of f, m, and n can be found from f by staying within the algebra, and m and n are the only external parameters required to obtain z_{mm} and z_{nn}.

7. BASIS REPRESENTATION OF CONVOLUTION

In the previous section, numerous macros were given in the image algebra. Most of these macros were simple, in that their representations in terms of the fundamental operations were almost obvious. In this section, one of the many more sophisticated image operations will also be given in terms of the algebra. The crucial operation of convolution has been chosen for this representation.

Let f and h be images with finite domains; that is,

$$f,\ h \in Y = \bigcup_{\substack{A \subset Z \times Z \\ \text{card } A < \infty}} R^A$$

Recall that the convolution of f and h is denoted by f*h. The representation of convolution in terms of the fundamental operations, and in terms of previous macros, proceeds in three steps:

a. First find the 180° rotation of h using the macro N^2; translate the result by (i,j), and multiply the resulting image by f, using macro $\mathcal{M}$, and then translate again, this time by (k,n), to obtain

$$u^{ij}(k,n) = T(\mathcal{M}(T(N^2(h),i,j),f),k,n)$$

b. Next sum all the $u^{ij}(k,n)$ using basis type addition $\oplus$, and then, from this sum, select the image at {(i,j)} utilizing the macro $\mathcal{S}$. Call this quantity g^{ij}. Hence,

$$g^{ij} = \mathcal{S}\left(\sum_{k,n=-\infty}^{\infty} u^{ij}(k,n), \{(i,j)\} \right)$$

At any fixed pixel, only a finite number of non-empty images $u^{ij}(k,n)$ are involved, and, therefore, convergence need not be discussed for the seemingly infinite sum above.

c. Finally, extend all the g^{ij} together, using the macro $\mathcal{E}$ to form the desired convolution f*h:

$$f * h = \mathcal{E}(g^{00}, \mathcal{E}(g^{01}, \mathcal{E}(g^{0-1}, \ldots)))$$

Notice that only a finite number of extension operations need be employed above because f and h have finite domains. An example will be given to illustrate the steps involved. This same problem was given in the following example.

Example:

Let the image f be given by

f =

2			
1	1	3	
0	2	4	
	0	1	2

and the image h by

$$h = \begin{array}{r|c|c|c|}
\hline
2 & & & \\ \hline
1 & -1 & 3 & \\ \hline
0 & 2 & 4 & \\ \hline
 & 0 & 1 & 2
\end{array}$$

The convolution of f and h will now be found, and each step will be illustrated, utilizing the above images. Rotate h by 180° and let the result be u:

$$u = N^2(h) = \begin{array}{r|c|c|c|c|}
\hline
1 & & & & \\ \hline
0 & & 4 & 2 & \\ \hline
-1 & & 3 & -1 & \\ \hline
-2 & & & & \\ \hline
 & -2 & -1 & 0 & 1
\end{array}$$

Next multiply f and u to obtain α:

$$\alpha = \mathcal{M}(f,u) = \begin{array}{r|c|c|c|}
\hline
1 & & & \\ \hline
0 & & 4 & \\ \hline
-1 & & & \\ \hline
 & -1 & 0 & 1
\end{array}$$

Find all translates of α and add them

$$\ldots \oplus \alpha_{01} \oplus \alpha_{10} \oplus \ldots = B = 4$$

This gives the image B (which is identically equal to 4 everywhere).

Select from B the image g^{00} consisting of the value at (0,0), $g^{00} = \mathcal{S}[B,\{(0,0)\}]$. A translation of the rotated image given previously will be performed, and many of the steps repeated. So translate u one unit to the right; thus $u_{10} = T(u,1,0)$.

Next, multiply f and u_{10} to obtain the image s, where $s = \mathcal{M}(f,u_{10})$:

2			
1			
0	8	8	
	0	1	2

Find all translates of s and add them together to obtain:

$$t = \ldots \oplus s_{01} \oplus s_{10} \oplus \ldots$$

Finally, select the image g^{10} which consists of the grey value of t at (1,0); $g^{10} = \mathcal{S}[t,\{(1,0)\}]$

2			
1			
0		16	
	0	1	2

Another translation of the rotated image will be conducted and again many of the steps will be represented. Translate u by two units to the right to give u_{20}:$u_{20} = T(u,2,0)$. Next multiply f and u_{20} to get Υ: $\Upsilon = \mathcal{M}(f,u_{20})$

2				
1				
0		16		
	0	1	2	3

Find all translates of Υ and add them together to give

$$\partial = \ldots \oplus \Upsilon \oplus \Upsilon_{01} \oplus \Upsilon_{10} \oplus \ldots = 16$$

Select from ∂ the image g^{20} made up of the value of ∂ on (2,0). $g^{20} = \mathcal{S}(\partial, \{(2,0)\})$. A large portion of the procedure is again repeated. Translate u one unit to the right and one unit up: $u_{11} = T(u,1,1)$. Next multiply f and u_{11} to get $r = \mathcal{M}(f, u_{11})$

2			
1	4	6	
0	6	-4	
	0	1	2

Find and add all the translates of r together to obtain

$$v = \ldots r_{0-1} \oplus r \oplus r_{01} \oplus r_{10} \oplus r_{11} \oplus \ldots = 12$$

Select the image g^{11}, consisting of the grey value of v at (1,1): $g^{11} = \mathcal{S}(v, \{(1,1)\})$

2			
1		12	
0			
	0	1	2

The procedure is again repeated. Translate u one unit up: $u_{01} = T(u,0,1)$. Multiply f and u_{01} to obtain the image χ. $\chi = \mathcal{M}(f, u_{01})$

2			
1	2		
0	-2		
	0	1	2

Find all translates of x and add them together to obtain

$$y = \ldots \oplus x \oplus x_{01} \oplus x_{11} \oplus \ldots = 0$$

Select the image g^{01}, consisting of the value of y at (0,1). $g^{01} = \mathcal{S}(y,\{(0,1)\})$

2			
1	0		
0			
	0	1	2

The procedure is repeated again. Translate u two units up: $u_{02} = T(u,0,2)$. Next multiply f and u_{02} giving w: $w = \mathcal{M}(f,u_{02})$

2			
1	-1		
0			
	0	1	2

Find all translates of w and add them together to yield the image

$$e = \ldots \oplus w \oplus w_{01} \oplus \ldots = -1$$

Select the image g^{02} consisting of the grey value at (0,2) from e. $g^{02} = \mathcal{S}(e,\{(0,2)\})$.

The procedure is repeated for a final time. Translate u two units to the right and two units up. $u_{22} = T(u,2,2)$. Multiply f and u_{22} giving the image d:

$$d = \mathcal{M}(f,u_{22}) =$$

2			
1		9	
0			
	0	1	2

Find all translates of d and add them together to get

$$\eta : \eta = \ldots \oplus d \oplus d_{01} \oplus \ldots = 9$$

Select the image g^{22}, consisting of the grey value at (2,2) from $\eta : g^{22} = \mathcal{S}(\eta, \{(2,2)\})$ gives the image g^{22}. The last step is to extend all g^{ij} together to obtain the desired convolution of f and h.

$$f * h = \mathcal{E}(g^{00}, \mathcal{E}(g^{10}, \mathcal{E}(g^{20}, \mathcal{E}(g^{11}, \mathcal{E}(g^{01}, \ldots\ldots)))))$$

and so on.

8. CHARACTERIZING THE MACRO OPERATIONS

The purpose of this section is to provide several ways of categorizing the macro operations. This characterization will involve objective, as well as subjective, attributes of the operations, in addition to mathematical and heuristic criteria. This characterization is useful for knowledge base systems involving the imaging algebras, as well as for autonomous image processing algorithm development.

First and foremost, any macro operation can be grouped according to its arity; that is the number of operands in its defining definition. Further classification is given by specifying the type of inputs utilized and the output sort which results. For the input, the order of the operands is also of prime concern.

Example

Referring to the previous sections, it is seen that a (relational) data base could be established involving the above type of information. An instance of this schema is given in Table C-2.

TABLE C-2. OPERATOR SYNTAX

Name of operator	operation	Arity	Domain Types arguments				Output Sort
			1	2	3	4	
Addition (fundamental)	⊕	2	image	image	-	-	image
Translation	T	3	image	integer	integer	-	image
Domain Finder	K	1	image	-	-	-	subset of ZxZ
Existential	E	2	real	subset of ZxZ	-	-	image
Unity image	1_A	0	-	-	-	-	image
Subtraction	⊖	1	image	-	-	-	image

As stated in this table, Addition is a binary operation, since its arity is 2 with both inputs being image, and with output also an image.

A polyadic graph (Figure C-5) is another useful way of representing the information contained within the flat file illustrated in Table C-2. This graph involves arrows with many tails corresponding to the arity of the operator. The head of the arrow points to the sort of output, which is indicated by using an oval containing the type of output. Each tail is also attached to an oval containing the sort of input utilized in the operation. For operators involving more than one sort of input, a slash mark is given to the tail to indicate the order within the operation. This is illustrated in Figure C-5.

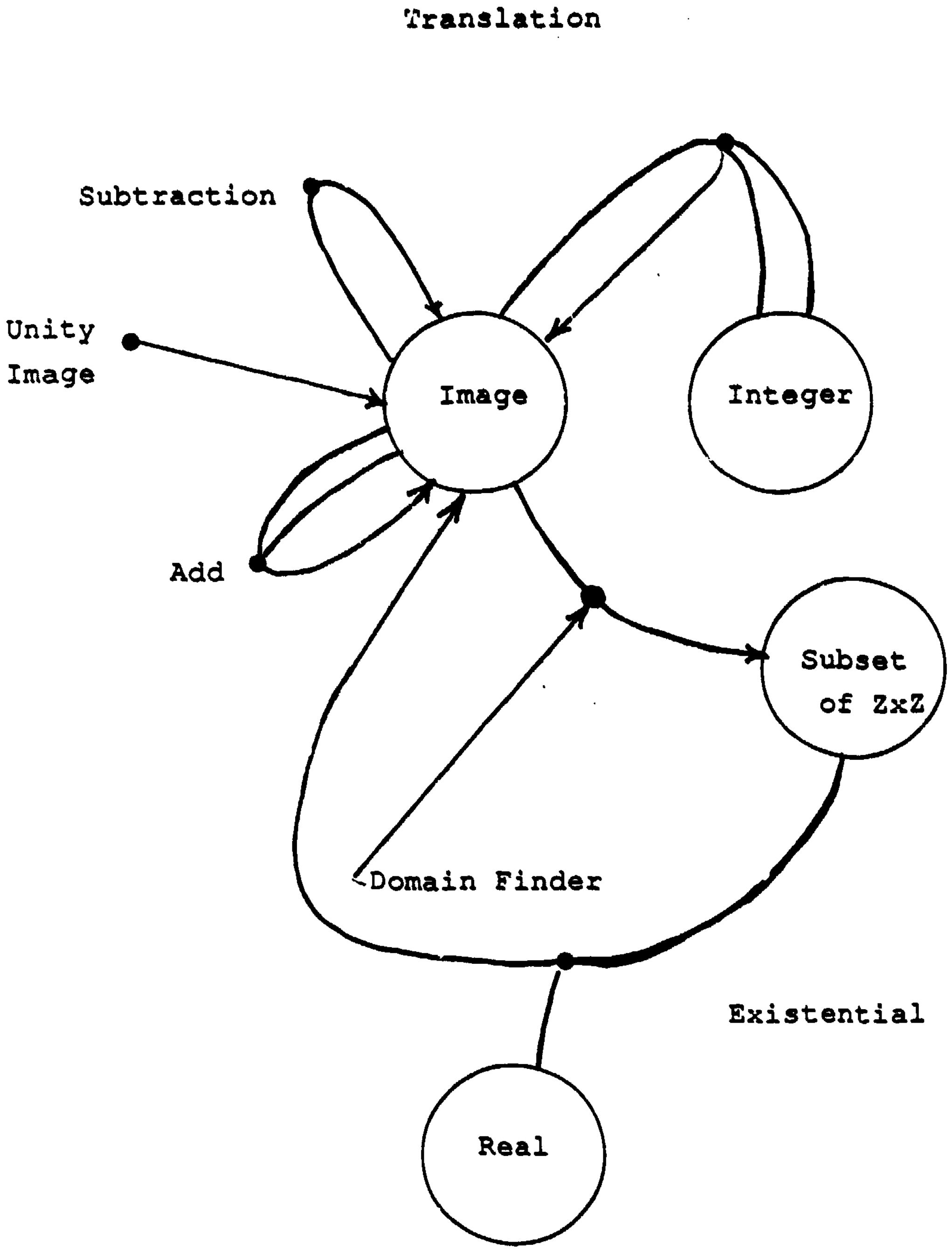


Figure C-5. Polyadic Graph

Equivalence classes of macro operations are established, utilizing this recording system. The partitioning procedure is of prime importance in syntax specification and program correctness.

A distinct way of partitioning and therefore characterizing these operations involves the nature of the function. Every operation described herein involves a (digital) image among its inputs or as an output. This motivates the following terminology. A macro operation is said to be an image creation macro or a creation macro if only the output involves an image. A macro operator is said to be an image transformation when both the output and input involve images. Finally, when only the input of a macro involves an image, then the macro is said to be a parameter determination macro.

Example

The translation, rotation and division operations are all transformation macros. The existential operation as well as the zero image are image creation macros. Finally, the domain finder and grey value determiner are both parameter determiner macros. In the former case, the parameter is a subset of ZxZ and, in the latter, it is a real number.

As indicated above, parameter determination macros are further broken down by specifying the type of parameter which is measured. In a more arbitrary fashion, transformation macros are further characterized: A transformation macro is said to be domain increasing if there exist image operands $f_1, f_2, \ldots, f_n$ in addition to other possible inputs, and for $i = 1, \ldots, n$ the cardinality of $K(f_i)$ < cardinality of the output domain.

Example

The fundamental addition operation is a domain increasing transformation since

$$1_{\{(0,0)\}} \oplus 0_{\{(1,0)\}} = g$$

$$g = \begin{array}{|c|c|c|}\hline 1 & & \\ \hline 0 & 1 & 0 \\ \hline & 0 & 1 \\ \hline \end{array}$$

and card $g = 2 >$ card $1_{\{(0,0)\}} =$ card $0_{\{(1,0)\}}$.

The fundamental division tranformation $\oplus$, translation T, and the subtraction macro $\ominus$ are not increasing.

In a similar manner, a transformation macro is said to be domain decreasing if there exist image operands $f_1, f_2, \ldots, f_n$, in addition to other possible inputs (if any), such that, for $i = 1, \ldots, n$, the cardinality $K(f_i) >$ cardinality of the domain of the output $i = 1, 2, \ldots, n$.

Example

The fundamental division operation is domain decreasing since $\oplus 0_{\{(0,0)\}} = \varnothing$ and card $0_{\{(0,0)\}} = 1$, whereas cardinality of $\varnothing = 0$ Notice that the selection macro $\mathcal{S}$ and the addition macro are both domain decreasing. Both the fundamental operations of translation and addition are not domain decreasing.

It should be noted that there exist operations which are both domain increasing and domain descreasing as the following example exemplifies.

Example

Consider the divide type of macro Q where Q: X x X → X and

$$Q(f,g)(x,y) = \begin{cases} f(x,y)/g(x,y) & g(x,y) \neq 0,\ (x,y) \in A \cap B \\ f(x,y) & (x,y) \in A - B \\ 1\ /\ g(x,y) & g(x,y) \neq = 0 \text{ and } (x,y) \in B - A \\ \text{undefined} & \text{otherwise} \end{cases}$$

with $A = K(f)$ and $B = K(g)$. It follows that

$$Q(f,g) = f \odot (\oplus g) = f \oplus g;$$

in any case, using $Q(1_{\{(0,0)\}}, 1_{\{(1,0)\}}) = 1_{\{(0,0,(1,0))\}}$

1			
0	1	1	
	0	1	2

shows that Q is domain increasing, while $Q(0_{\{(0,0)\}}, 0_{\{(0,0)\}}) = \emptyset$ shows that it is domain increasing.

Another important concept is that of domain stability. A transformation macro is said to be domain stable if it is not domain increasing or domain decreasing. Specifically, domain stability means that, for all possible sets of input operands $f_1, f_2, \ldots, f_n$ and all other sets of inputs (if any), there exists an input image f_i such that cardinality of $K(f_i)$ = cardinality of the output.

Example

Notice that the translation, rotation and flip operations are all domain stable, as is the scalar multiplication macro.

Two special types of domain stable transformation macros will now be defined. Both happen to be illustrated in the above example. The first type of domain stable transformation is called a rigid transformation. Intuitively, this type of transformation takes an image in X and only moves it to yield another image in X--no operation on the grey values is performed. Normally only domain induced operations are utilized in creating these types of operations. More rigorously, Q, is a domain stable transformation said to be rigid means that Q is expressible as a term involving only the fundamental operations of translation T, 90° rotation N and diagonal flip D; that is, Q is definable under function composition, utilizing only the operations T, N, and D.

Example

Consider a transpose type operation Q on images, where

$$Q: \quad X \rightarrow X \text{ and } Q(f)(x,y) = f(1-y,2-x).$$

In particular, if

f =

2	3	-1		
1	2	4		
0	1	7		
	0	1	2	3

it follows that

Q(f) =

1	3	2	1	
0	-1	4	7	
	0	1	2	3

Furthermore, Q(f) = T(D(f),1,2). As a consequence, the transpose operation above is a rigid transformation.

The second type of domain stable operation is called domain invariant. It is defined for operators requiring a single image input and, as the name suggests, this type of transformation macro must have an image output whose domain equals the domain of the input image.

Example

Let us find all domain stable transformations which are at the same time rigid and domain invariant. The first thing to notice is that, by repeatedly employing either D or N to a given image, at most only eight different images result (including the original). This discussion is related to the previous presentation on the octic group. An instance of the eight possibilities is given below in Figure C-6.

f

2				
1	1			
0	1	1	1	
	0	1	2	3

N(f)

2		1		
1		1		
0	1	1		
	-1	0	1	2

$N^2(f)$

1				
0	1	1	1	
-1			1	
	-2	-1	0	1

$N^3(f)$

0		1	1	
-1		1		
-2		1		
	-1	0	1	

D(f)

0	1	1		
-1		1		
-2		1		
	-1	0	1	2

N(D(f)) = F(f)

0	1	1	1	
-1	1			
-2				
	0	1	2	

D(N(f)) = V(f)

1			1	
0	1	1	1	
-1				
	-2	-1	0	

$N^2(D(f))$ = U(f)

2	1			
1	1			
0	1	1		
	0	1	2	

Figure C-6. Optic Group

Furthermore, it should be noticed that the translation operation is of no value in mapping any of the images in Figure C-6 (excluding f) into an image with the same domain as f. Translations applied anywhere will yield images with the same basic shapes as those illustrated in Figure C-6. This follows b observing that there always exists integers i and j, such that

$$T(N(T(f,p,q)),i,j) = N(f),$$

and there exists integers i', and j' such that

$$T(D(T(f,p,q)),i',j') = D(f).$$

This shows that utilization of translation in the midst of employing D and T will be of no value; that is, the only images obtainable under function composition involving T, D and N are the eight images depicted in Figure C-6, along with their translates. It follows that the only rigid domain invariant transformation macro is the identity operation I, where I(f)(x,y) = f(x,y).

The immediate discussion will be concluded by way of an example illustrating a domain stable transformation macro which is neither domain invariant nor rigid.

Example

Consider the macro operation Q where X → X and Q(f)(x,y) = 2f(y,x). Then

$$Q(f) = (2 \triangle U(f))$$

For instance, using f as given in the example yields the image.

$$Q(f) = \begin{array}{c|c|c|c} 2 & 2 & & \\ \hline 1 & 2 & & \\ \hline 0 & 2 & 2 & \\ \hline & 0 & 1 & 2 \end{array}$$

and so card K(f) = card K(Q(f)) which shows that Q is a domain stable transformation. However, by Example, Q(f) is not obtainable as a term from f using N,D and T, and so Q is not a domain invariant transformation, and it certainly is not rigid. This ends the example.

An additional, perhaps obvious, way of characterizing the macro operation is by name. Names are often indicative of purpose. For instance, all addition type operations should be grouped together. In this grouping, there would appear: the fundamental addition operation $\oplus$; the intersection type addition macro, the Minkowski addition operation $\boxplus$, etc. A somewhat similar type of characterization would arise by grouping macros according to purpose. It should be mentioned that actual physical grouping is not what is intended here; rather, it is the (logical) linking together of the information. Conventional data base-data structure techniques such as linked lists or relational structures could be used for this purpose. Additional so-called AI techniques such as semantic net or frame structures would also be appropriate. An example of a semantic net structure incorporating some of the information illustrated in this section is given below.

Example

Consider the transformation macros, a, $\oplus$, $\textcircled{\vee}$, T. Then a semantic net representing some properties of these operations is given in Figure C-7.

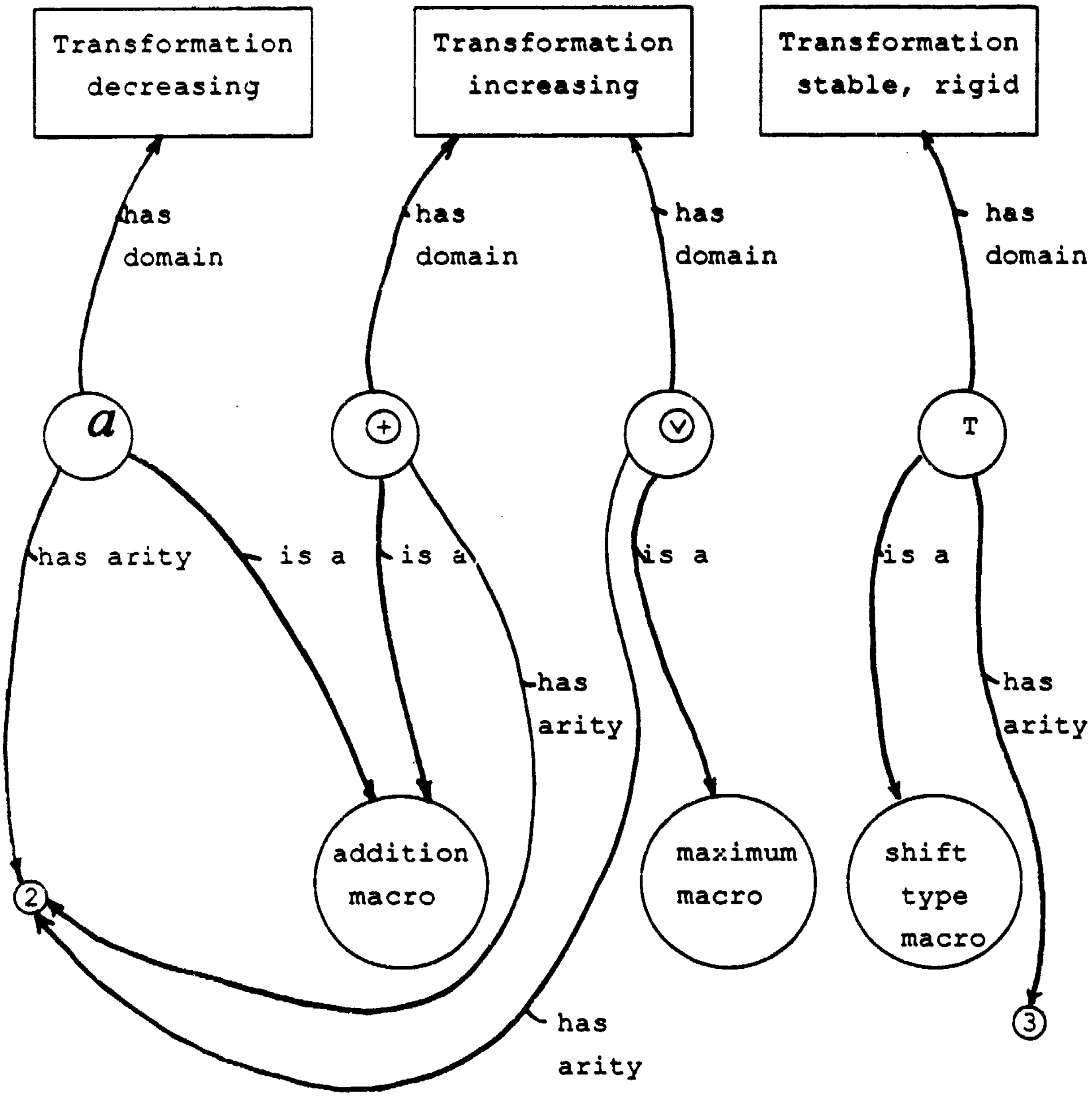


Figure C-7. Semantic Net Diagram

The same information given in the Example is again provided, using a relation data base flat file in Table C-3. The benefit of the latter approach is immediate.

Example

TABLE C-3. FUNCTION TYPE

Macro Operation Symbol	Type of Operation	Arity	Nature of Function Creation	Transformation Type	Parameter Type
a	Add	2		Decreasing	-
$\oplus$	Add	2		Increasing	-
$\textcircled{\vee}$	Maximum	2		Increasing	-
T	Translate	3		Stable-rigid	-

Other types of groupings are of equal importance to the types mentioned above. For instance, grouping together: arithmetic operations, or logical-lattice type operations, or morphological operations, or even data base-type operations, such as the extension macro operation $\mathcal{E}$ and the selection macro $\mathcal{S}$.

9. MATHEMATICAL INDUCEMENT OF BASIS OPERATORS IN THE IMAGING ALGEBRA

Earlier, it was mentioned that several of the fundamental operators in the image algebra were either range or domain induced operations. However, these notions were not defined.

In this section, the meaning of range inducement will be given and illustrated. This is followed by the definition of domain inducement. It will then be shown that the fundamental operations of translation, rotation and reflection are all domain induced.

It was previously seen that one of the principal reasons for dealing with induced operations is the simplicity in which polynomial term specifications of image processing operations and algorithms can be determined. An additional reason for utilizing induced operations is in variety specification. Here a very important theorem is proven, showing that range induced operations commute with domain induced operations. A consequence of this result is that any of the domain induced fundamental operations (mentioned above) commute with any of the other fundamental transformation operations.

The operator R is said to be range induced by r if:

a. r is an n+k ary function $n \geq 1, k \geq 0$, with
$r: B^n \times B_1 \times \ldots \times B_k \rightarrow B$

b. $R: [\bigcup_{H \subset A} B^H]^n \times B_1 \times \ldots \times B_k \rightarrow \bigcup_{H \subset A} B^H$

c. for any f_i in $\bigcup_{H \subset A} B^H$, $i = 1,2 \ldots, n$ and b_j in B_j ,
$j = 1,2, \ldots, k$

it follows that

$$R\,(f_1, f_2, \ldots, f_n, b_1, b_2, \ldots, b_k)\ (x) = r(f_1(x), \ldots, f_n(x), b_1, \ldots, b_k)$$

for all the points where each $f_i(x)$ is defined at the same time. If every f_i is defined on the same domain; that is $f_i \in B^A$ for i

= 1,2,...,n, if

$$R : (B^A)^n \times B_1 \times \ldots \times B_k \rightarrow B^A$$

and if R is range induced, then

$$R (f_1 ,f_2,...,f_n,b_1, \ldots ,b_k) (x) = r(f_1(x),...,f_n(x),b_1,...,b_k)$$

for every x in A. For the more general case cited in the definition, the equality need only hold in the intersection of the domains of the functions involved. The intersection of these domains herein is called the common region.

Example

Notice that the unary operation of subtraction, r = -, is such that

$$- : R \rightarrow R$$

As a result, the macro subtraction image operation R = $\ominus$, where

$$\ominus : \bigcup_{H\subset Z\times Z} R^H \rightarrow \bigcup_{H\subset Z\times Z} R^H$$

in the imaging algebra is range induced by - , since

$$[\ominus(f)](u,v) = -f(u,v)$$

Example

Consider the binary addition operation, r = +,

$$+: R \times R \rightarrow R$$

The macro imaging addition operation for images is range induced by +. This follows since

$$a : \bigcup_{H\subset Z\times Z} R^{H} \times \bigcup_{H\subset Z\times Z} R^{H} \to \bigcup_{H\subset Z\times Z} R^{H}$$

and

$$a(f,g)(u,v) = f(u,v) + g(u,v)$$

for all pixels (u,v) in the intersection of the domain of f and g. Recall that a is not defined elsewhere. If the operator happened to be defined elsewhere in an arbitrary fashion, it still would be considered to be range induced.

Example

Again consider binary addition as in the last example. The fundamental addition operation $\oplus$ in the algebra is also range induced by +. This follows since

$$\oplus : \bigcup_{H\subset Z\times Z} R^{H} \times \bigcup_{H\subset Z\times Z} R^{H} \to \bigcup_{H\subset Z\times Z} R^{H}$$

and

$$(f \oplus g)(u,v) = f(u,v) + g(u,v)$$

for all points in the intersection of the domains K(f) of f and K(g) of g. Thus, the common region here is $K(f) \cap K(g)$. Recall that $(f \oplus g)(u,v)$ was (heuristically) defined as

$$(f \oplus g)(u,v) = f(u,v)$$

for $(u,v) \in K(f) - K(g)$ and

$$(f \oplus g)(u,v) = g(u,v)$$

for $(u,v) \in K(g) - K(f)$ and was undefined elsewhere. The way

$\oplus$ is defined outside the common domain of its operands f and g is irrelevant to the inducement process.

In an analogous fashion, it follows that the fundamental multiplication operation $\odot$ is range induced by the multiplication of the reals, and the fundamental maximum operation $\textcircled{\vee}$ in the image algebra is induced by the maximum operation $\vee$ in the real lattice.

Example

Consider the reciprocal partial operation on the reals. Here

$$+ : R_0 \to R_0$$

where $R_0 = (-\infty,0)\cup(0,\infty)$. This operator range induces a type of division operation $\boxplus$ on a subset of images where

$$\boxplus : \bigcup_{H\subset Z\times Z} R_0^H \to \bigcup_{H\subset Z\times Z} R_0^H$$

Therefore, $\boxplus$ only operates on images which have arbitrary domains K(f) and no grey values equal to zero. Notice that

$$\boxplus(f)(u,v) = \frac{1}{f(u,v)}$$

for (u,v) in K(f) and is undefined elsewhere. Notice that the fundamental division operation $\oplus$ in the imaging algebra is an extension of the division operation $\boxplus$. Furthermore, on the extended domain of $\oplus$ that is, for those functions f in $\bigcup_{H\subset Z\times Z} R^H$ which have at least one zero grey value, the operators $\boxplus$ and $\oplus$ provide the same answer when applied to f restricted to the domain equal to the support region of f. Because of this, we say that $\oplus$ is a support extended extension of a range induced operation induced by +.

The function D, where

$$D: \bigcup_{H\subset A} B^H \times A_1 \times \ldots \times A_{n'-1} \rightarrow \bigcup_{H\subset A} B^H$$

is said to be domain induced by the operation d where

$$d: \; A \times A_1 \times \ldots \times A_{n'-1} \rightarrow A, \text{ if}$$

$$D\,[f, a_1, a_2, \ldots, a_{n'-1}](x) = f[d(x, a_1, a_2, \ldots, a_{n'-1})]$$

Example

Let d be the real vector type (binary) subtraction of two vector tuples, thus:

$$d: \; (Z\times Z) \times (Z\times Z) \rightarrow (Z\times Z),$$

where

$$d((u,v),\; (i,j)) = (u-i, v-j).$$

This function induces (up to isomorphism) the fundamental operation of image translation; indeed,

$$D = T, \text{ where } \; D: \bigcup_{H\subset Z\times Z} R^H \times (Z\times Z) \rightarrow \bigcup_{H\subset Z\times Z} R^H$$

Here $n' = 2$ and so

$$A_1 = A = (Z\times Z)$$

and B = R. Thus,

$$\bigcup_{H\subset Z\times Z} R^H$$

is the set of all images described previously, and the translation operator is

$$T(f,(i,j))(u,v) = f(u-i,v-j)$$

Example

Suppose

$$d: (Z \times Z) \rightarrow (Z \times Z)$$

where

$$d((u,v)) = (v,-u)$$

Then d is a (unary) operation on vectors. This operation corresponds to the matrix (rotation) operation.

$$\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

on the column vector

$$\begin{pmatrix} u \\ v \end{pmatrix}$$

Notice that $n' = 1$ in this case, and if B = R and D = N, then the fundamental operation of counter-clockwise 90° rotation is induced. Hence,

$$N: \bigcup_{H \subset Z \times Z} R^H \rightarrow \bigcup_{H \subset Z \times Z} R^H$$

where

$$N(f)(u,v) = f(d(u,v)) = f(v,-u)$$

Example

Define d in the above example as

$$d((u,v)) = (-v,-u)$$

Then this operation corresponds to the idempotent matrix operation

$$\begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix}$$

on the column vector

$$\begin{pmatrix} u \\ v \end{pmatrix}$$

The domain induces operation on images is the fundamental diagonal flip operation **D** where

$$\mathbf{D}(f)(u,v) = f(-v,-u)$$

Among the numerous benefits of employing an algebra whose fundamental operations are induced is the fact that range induced and domain induced operators commute. The precise statement of this fact is given below in Theorem 1. Utilizing this theorem along with the commutativity of domain induced operations and the extension operation provides more powerful theorems (such as Theorem 1) on the commutativity of operators in the image algebra.

Theorem 1:

If **R** is a range induced operation and **D** is a domain induced operator, then these operations commute on the common region.

Comment:

Recall that the common region consists of all points in the intersection of the domains of the function $f_1, f_2, \ldots, f_n$ used in range inducement. In the common region,

$$\mathbf{R}(f_1, f_2, \ldots, f_n, b_1, b_2, \ldots, b_k)\ (x) \\ = r(f_1(x), \ldots, f_n(x), b_1, \ldots, b_k)$$

holds true. No restrictions are needed in domain inducement: here

$$\mathbf{D}(f, a_1, a_2, \ldots, a_{n'-1})(x) = f(d(x, a_1, \ldots, a_{n'-1})$$

The proof will be given with the help of the diagram (Figure C-8) given below. Specifically, it will be shown that this diagram commutes; that is, by traversing it first to the right and then down (paths I,II) gives the same results as if it were traversed first going down and then to the right (paths III, IV).

$$\begin{tikzcd}
(f_1, f_2, \ldots, f_n) \in \bigcup_{H \subset Z \times Z} R^{Hn} \arrow[rr, "\text{Path I}"] \arrow[dd, "\text{Path III}"'] & & f \in \bigcup_{H \subset Z \times Z} R^{H} \arrow[dd, "\text{Path II}"] \\
\mathbf{D} \times \mathbf{D} \times \ldots \times \mathbf{D} & & \mathbf{D} \\
(f_1^*, f_2^*, \ldots, f_n^*) \in (\bigcup_{H \subset Z \times Z} R^{H})^n \arrow[rr, "\text{Path IV}"'] & & f^* \in \bigcup_{H \subset Z \times Z} R^{H}
\end{tikzcd}$$

Figure C-8. Commuting Diagram of Domain and Range Commutativity

When f^* and h are restricted to the common region, we will show that they are equal; that is, using

$$f = R(f_1, f_2, \ldots, f_n, b_1, b_2, \ldots, b_k)$$

and

$$h = D(f, a_1, a_2, \ldots, a_{n'-1})$$

along with

$$f_i^* = D(f_i, a_1, \ldots, a_{n'-1}) \quad i = 1, 2, \ldots, n$$

and

$$f^* = R(f_1^*, f_2^*, \ldots, f_n^*, b_1, b_2, \ldots, b_k)$$

we will show that

$$f^*(x) = h(x)$$

for x in the common region.

Proof:

By traversing the diagram above along path I and using range inducement gives

$$f(x) = r(f_1(x), f_2(x), \ldots, f_n(x), b_1, \ldots, b_k)$$

then using path II along with domain inducement yields the fact that

$$h(x) = r(f_1(d(x, a_1, \ldots, a_{n'-1})), \ldots, f_n(d(x, a_1, \ldots, a_{n'-1})), b_1, \ldots, b_k)$$

This, of course, holds in the common region mentioned above.

Next, if the diagram is traversed first along path III and domain inducement is used n times, then,

$$f_i^*(x) = f_i(d(x,a_1,a_2,\ldots,a_{n'-1}))\quad i = 1,2,\ldots,n$$

Then traversing path IV and using range inducement gives

$$f^* = r(f_1^*(x),f_2^*(x),\ldots,f_n^*(x),b_1,\ldots,b_k)$$

If, in this expression,

$$f^*(x) = f_i(d(x,a_1,\ldots,a_{n'-1}))$$

is substituted, it is seen that $f^*(x) = h(x)$, thereby concluding the proof.

Corollary:

Let **R** be a range induced operation, yielding an image whose domain is the intersection of the domains of its image operands. If **D** is a domain induced operator, then **R** and **D** commute, along with

$$f_i^* = \mathbf{D}(f_i,a_1,\ldots,a_{n'-1})\quad i = 1,2,\ldots,n$$

and

$$f^* = \mathbf{R}(f_1^*,f_2^*,\ldots,f_n^*,b_1,b_2,\ldots,b_k)$$

we will show that

$$f^*(x) = h(x)$$

for x in the common region.

Proof:

By traversing the diagram above along path I and using range inducement gives

$$f(x) = r(f_1(x), f_2(x), \ldots, f_n(x), b_1, \ldots, b_k)$$

then using path II along with domain inducement yields the fact that

$$h(x) = r(f_1(d(x, a_1, \ldots, a_{n'-1})), \ldots, f_n(d(x, a_1, \ldots, a_{n'-1})), b_1, \ldots, b_k)$$

This, of course, holds in the common region mentioned above.

Example

If the range induced operator **R** is $\boldsymbol{a}$, the add macro, then $\boldsymbol{a}$ commutes with any domain induced image operator **D**. So, for instance, $\boldsymbol{a}$ commutes with translation T, 90° rotation N, and reflection D. Thus,

$$T(\boldsymbol{a}(f,g),i,j) = \boldsymbol{a}(T(f,i,j), T(g,i,j)),$$

$$N(\boldsymbol{a}(f,g)) = \boldsymbol{a}(N(f), N(g))$$

and

$$D(\boldsymbol{a}(f,g)) = \boldsymbol{a}(D(f), D(g))$$

Similar results can be given for the multiplication macro $\boldsymbol{M}$, as well as the higher macro $\boldsymbol{H}$, lower macro £ and numerous other macros.

One of the immediate goals will be to show that any domain induced image operation commutes with the fundamental binary range induced operators in the image algebra. In order to show this result, it will be shown that these operators are comprised of more trivial operators, all of which do commute with any domain induced operation. First it is noticed that the three fundamental binary range induced operations are representable as terms using the composition macro and certain other range induced operations discussed in the previous example. This result is given in Theorem 2.

Theorem 2:

The fundamental operations of addition, multiplication and maximum have the respective representations:

$$\begin{aligned} f \oplus g &= \varepsilon(\varepsilon\ (\boldsymbol{a}\ (f,g),f),g) \\ f \odot g &= \varepsilon(\varepsilon\ (\boldsymbol{M}\ (f,g),f),g) \\ f \circledvee g &= \varepsilon(\varepsilon\ (\boldsymbol{H}\ (f,g),f),g) \end{aligned}$$

The proof of these representations follow directly from the definition.

Next it will be noticed that ε commutes with any domain induced operation.

Theorem 3:

If **D** is any domain induced operation in the image algebra and ε is the composition operator, then these operations commute; that is,

$$\varepsilon(\ \mathbf{D}(f,a_1,a_2,\ldots,a_{n'-1}),\mathbf{D}(g,a_1,a_2,\ldots,a_{n'-1}))$$
$$= \mathbf{D}(\varepsilon(f,g),a_1,a_2,\ldots,a_{n'-1})$$

Proof:

The diagram (Figure C-9) below will be shown to commute. Here it is assumed that

$f \in R^A$ and $g \in R^B$ with
$A,B \subset Z{\times}Z$ and $a_i \in A_i$, $i = 1,2,\ldots,n'-1$.

$$\begin{tikzcd}
(f,g) \in R^A \times R^B \arrow[r] \arrow[d, "\mathbf{D}\times\mathbf{D}"'] & b \in \bigcup_{H\subset Z\times Z} R^H \arrow[d, "\mathbf{D}"] \\
(l,m) \subset \bigcup_{H\subset Z\times Z} R^H \times \bigcup_{H\subset Z\times Z} R^H \arrow[r] & \bigcup_{H\subset Z\times Z} h^H_R
\end{tikzcd}$$

Figure C-9. Commuting Diagram

First, traversing this diagram from left to right and then down gives

$$h = \mathbf{D}(b,a_1,a_2,\ldots,a_{n'-1}) = \mathbf{D}(\varepsilon(f,g),a_1,a_2,\ldots,a_{n'-1})$$

where

$$\varepsilon(f,g)(v) = \begin{cases} f(v) & v \in A \\ g(v) & v \in B - A \end{cases}$$

and v denotes the two tuple (x,y). Using domain inducement gives

$$h(v) = \begin{cases} f(d(v,a_1,a_2,\ldots,a_{n'-1})) \text{ if } d(v,a_1,\ldots,a_{n'-1}) \in A \\ g(d(v,a_1,a_2,\ldots,a_{n'-1})) \text{ if } d(v,a_1,\ldots,a_{n'-1}) \in B - A \end{cases}$$

Next, traversing the diagram down and then to the right gives

$$p = \varepsilon(l,m) = \varepsilon(\mathbf{D}(f,a_1,a_2,\ldots,a_{n'-1}),\mathbf{D}(g,a_1,a_2,\ldots,a_{n'-1}))$$

Using the domain inducement gives

$$l(v) = \mathbf{D}(f,a_1,a_2,\ldots,a_{n'-1})(v) = f(d(v,a_1,\ldots,a_{n'-1}))$$

and

$$m(v) = \mathbf{D}(g,a_1,a_2,\ldots,a_{n'-1})(v) = g(d(v,a_1,\ldots,a_{n'-1}))$$

Therefore,

$$p(v) = h(v)$$

<u>Theorem 4:</u>

Any of the binary range induced fundamental image operations commute with any domain induced image operation.

<u>Proof:</u>

The proof will be conducted for addition; the proof of other binary range induced operators are identical. It will be shown that, for any domain induced operation

$$\mathbf{D}\ (f \oplus g) = \mathbf{D}\ (f) \oplus \mathbf{D}\ (g)$$

By employing the representation given in Theorem 2, we have

$$\mathbf{D}\ (f \oplus g) = \mathbf{D}\ (\varepsilon(\varepsilon(\mathcal{A}(f,g),f),g))$$

Utilizing the results of Theorem 3 gives

$$\mathbf{D}(f \oplus g) = \varepsilon(\mathbf{D}(\varepsilon(a(f,g),f)),\mathbf{D}(g))$$

Applying this same theorem again gives

$$\mathbf{D}\ (f \oplus g) = \varepsilon(\varepsilon(\mathbf{D}(a(f,g)),\mathbf{D}(f)),\mathbf{D}(g))$$

Lastly, apply Theorem 1 to obtain

$$\mathbf{D}\ (f \oplus g) = \varepsilon(\varepsilon(a(\mathbf{D}(f),\mathbf{D}(g)),\mathbf{D}(f)),\mathbf{D}(g))$$

By again using Theorem 2, it is noticed that the right hand side of the above expression is

$$\mathbf{D}(f) \oplus \mathbf{D}(g)$$

thereby concluding the proof.

This section will be completed by showing that the fundamental division operation also commutes with any domain induced image operation.

Theorem 5:

The fundamental image operation of division $\oplus$ commutes with any domain induced image operator.

Proof:

Let $h = \oplus f$ and let $g = \mathbf{D}(h,a_1,\ a_2,\ldots,a_{n'-1})$

Also let

$$b = \mathbf{D}(f,a_1,\ldots,a_{n'-1})$$

and

$$g^* = \oplus b$$

It will be shown that $g = g^*$. Let $f \in R^A \subset Z \times Z$, and notice that

$$h(v) = \begin{cases} 1/f(v) & \text{whenever } f(v) \neq 0 \text{ in } A \\ \text{undefined} & \text{otherwise} \end{cases}$$

Here again, v denotes the pixel (x,y). Therefore,

$$g(v) = \begin{cases} \mathbf{D}(\frac{1}{f(v)}, a_1, a_2, \ldots, a_{n'-1}) & \text{when } f(v) \neq 0 \\ \text{undefined} & \text{elsewhere} \end{cases}$$

By using the domain inducement, we get

$$g(v) = \begin{cases} \frac{1}{f(d(v, a_1, a_2, \ldots, a_{n'-1}))} & \text{when } f(d(v, a_1, \ldots, a_{n'-1})) \neq 0 \\ \text{undefined} & \text{elsewhere} \end{cases}$$

On the other hand, using the domain inducement for b gives

$$b(v) = f(d(v, a_1, a_2, \ldots, a_{n'-1}))$$

Finally, using the reciprocal operation provides the desired result:

$$g^*(v) = \begin{cases} \frac{1}{f(d(v, a_1, a_2, \ldots, a_{n'-1}))} & \text{when } f(d(v, a_1, \ldots, a_{n'-1})) \neq 0 \\ \text{undefined} & \text{elsewhere} \end{cases}$$

10. ON THE VARIETY OF THE IMAGE ALGEBRA

On the various laws and identities specified by equations hold in the algebra developed herein. Many of these identities occur due to the inducement process as was the case for the commutativity of range and domain induced operations seen in the last section.

Numerous laws such as the associative law, the distributive law, and so on are satisfied by various operators in the algebra. As a consequence, numerous sub-algebras exist within the image algebra. A few obvious sub-algebras within the image algebra will be noted below.

Under the fundamental addition $\oplus$ or multiplication $\odot$, the set of all images form a commutative monoid. This follows since the equational constraints given next are satisfied.

A1) Associative Law: $f_1 \oplus (f_2 \oplus f_3) = (f_1 \oplus f_2) \oplus f_3$

A2) Zero Law: $f \oplus \emptyset = \emptyset \oplus f = f$ ($\emptyset$ is the empty image.)

A3) Commutative Law: $f \oplus g = g \oplus f$

and

M1) Associative Law: $f_1 \odot (f_2 \odot f_3) = (f_1 \odot f_2) \odot f_3$

M2) Identity Law: $f \odot \emptyset = \emptyset \odot f = f$

M3) Commutative Law: $f \odot g = g \odot f$

These structures are not groups, since there is no inverse operation for either $\oplus$ or $\odot$.

There also is a sub-structure within the imaging algebra which is a join semi-lattice with a least element. This follows by using the set of all images X along with the fundamental maximum operation $\ovee$ since the following identities hold.

J1) Associative Law: $f \ovee (g \ovee h) = (f \ovee g) \ovee h$
J2) Commutative Law: $f \ovee g = g \ovee f$
J3) Idempotent Law: $f \ovee f = f$

J4) Least Element Law: $f \ovee \varnothing = f$ ($\varnothing$ is the empty image)

It obviously follows by symmetry, using the minimum macro $\owedge$, that there also exists a meet semi-lattice with greatest element within the algebra. However, it is easily seen that $(X, \ovee, \owedge)$ is not a lattice. This follows using the counter example given in the following example.

Example

The absorption law: $(f \ovee g) \owedge g = g$ need not hold in the imaging algebra. If $f = 0_{\{(0,0)\}}$ and $g = 1_{\{(1,0)\}}$ then

$(f \ovee g) \owedge g =$

1			
0	0	1	
	0	1	2

Many other sub-algebras exist within the image algebra. One very rich structure is that of a vector space. Specifically, if we let $V \subset X = \bigcup R^B$, $B \subset Z \times Z$, where $V = R^A$, $A \neq \varnothing$, then V is a vector space over the reals where image addition: $\oplus$ is the vector addition and the macro $\triangle$ is the scalar multiplication and the eight axioms hold:

V1) Associative Law For Vector Addition:

$x \oplus (y \oplus z) = (x \oplus y) \oplus z$

V2) Identity For Vector Addition: there exists a unique element 0_A in V called the zero image, such that

$$x \oplus (\ominus x) = 0_A \text{ for every image } x.$$

V3) Inverse For Vector Addition: for every x in V there exists a unique image $(\ominus x)$ in V, such that

$$x \oplus (\ominus\ x) = 0_{\lambda}$$

V4) Commutative Law For Vector Addition:

$x \oplus y = y \oplus x$

V5) Associative Law for Scalar Multiplication:

$ab \mathbin{\triangle} x = a \mathbin{\triangle} (b \mathbin{\triangle} x)$ (a,b are scalar)

V6) Identity for Scaler Multiplication:

$1 \mathbin{\triangle} x = x$, for every x in V

V7) Distributive Law For Vector Addition:

$a \mathbin{\triangle} (x \oplus y) = [a \mathbin{\triangle} x] \oplus [a \mathbin{\triangle} y]$

V8) Distributive Law For Scalar Addition:

$(a \oplus b) \mathbin{\triangle} x = [a \mathbin{\triangle} x] \oplus [b \mathbin{\triangle} x]$ (a,b are scalar)

We will conclude this section with another very rich structure within the image algebra. This structure involves two domain induced operations.

Consider any image f along with the fundamental operations of rotation N and reflection D. Let S be the set of all images obtained by successively employing the operator N and D to f. This structure is an octic group with generator elements N and D. A graph of this octic group is given in Figure C-6 involving image f and macro operations N^2,N^3,V, F, and U.

APPENDIX D

NAMES AND SAMPLES OF COLLECTED IMAGE PROCESSING TRANSFORMS

1. IMAGE PROCESSING TRANSFORM NAMES

Alpha Conditional Bisector

Array Grammars

Asynchronous Interaction

Background Subtraction

Bandwidth Compression Via Iterative Histogram Modification

Bernstein Polynomial Approximation

Best Plane Fit (BPF, Sobel, Roberts, Prewitt, Gradient)

Boundary Finder

Boundary Segmenter

Chain Code Angle Determiner

Closing (black and white)

Closure Operation

Connection Operator

Connectivity Number

Convex Hull

Convexity Number

Convolution Transform

Co-occurrence Matrix

Cumulative Angular Deviant Fourier Description

Cue Transform

Digitalization

Dilation (black and white)

Directional Gradient Transform

Discrete Cosine Transform

Discrete Fourier Transform

Discrete K-L Feature Selection
Discrete Picture Tranform
Edge Detection by Gradient
Erosion (black and white)
Fields Without Interaction (black and white)
Fourier Feature Normalization
Fourier Features
Frei-Chen Thresholding Strategy
Geometric Correction
Gibbs Ensemble (black and white)
Gradient Directed Segmentation
Gradient Edge Operators
Grey Scale Correction
Grey Scale Histogram
Grey Scale Transformation
Haar Transform (Haar Functions)
Hadamard Transform (Walsh Transform)
Heukel Edge Operator
Hierarchical Edge Detection
Histogram Equalization
Hit and Miss Transformation
Hotelling Transform (Karhunen-Loeve Transform)
Hough Method for Line Detection
Image Coding by DPCM
Intersection Function (black and white)
Kirsch Operator
Linear Erosion
Linear Filtering
Local Neighborhood Transform
L-U Decomposition
Magnitude Gradient Transform
Markov Random Field (black and white)
Medial Axis Skeleton
Minkowski Functionals
Moments of Silhouette

Morphological Covariance
Noise Removal by Smoothing
Opening (black and white)
Farallel Interactive Scene Labeling
Perimeter Estimation by Dilation
Planar Size Distribution
Pyramid
Projection Estimation by Dilation
Quad Trees (and Binary Trees)
Quantization
Random Field
Raster Tracking
Region Growing
Relaxation Labeling
Rotation Invariant Field (black and white)
Sequential Thinning
Serial Relaxation
Shape Grammars
Simple Boundary Segmenter
Size Criteria
Size Distribution in Length
Skeleton
Skiz Transform (Exoskeleton)
Slant Transform
Spoke Filter
Stacked-Image Data Structure
Straight Line Detection by Linear Filter
Subtraction of Laplacian
Superslice
Superspike
Syntactic Noise Reduction
Template Matching
Template Matching by Cross Correlation
Thickening
Thinning (Morphological)

Thresholding
Transform Encoding
Translation Transform
Tree Grammars
Tree Search Labeling Algorithm
Umbra Transform
Variable Sized Hexagons
Walsh Feature Representation
Zucker and Hammel Three Dimensional Edge Operators

2. SAMPLES OF THE IMAGE PROCESSING TRANSFORMS

a. Best Plane Fit (BPF)

(1) Classification:

Edge Detection

(2) Purpose and Methodology:

The BPF technique is used primarily for edge detection. It is employed on digital images by finding a plane which locally best approximates this image. Local relative to a pixel will be a neighborhood of a pixel; it may consist of three, four or sometimes more pixels. The criteria of Best will be to minimize some cost functional here, the Euclidean Norm. The coefficients of the plane are indicative of the presence of a gradient. This is determined by applying another functional.

(3) Mathematical Description:

Consider the 3 x 3 mask, illustrated below, for finding the gradient at the center pixel whose grey value is x_0. The grey value of the neighboring pixels is denoted by $x_1, x_2, \ldots, x_8$ and are also illustrated. The absolute location of the central pixel is $(i,j) \in Z \times Z$; therefore, its grey value $x\ (i,j) = x_0$. Similarly, for the neighboring pixels we have

$$\begin{array}{ccc} x_2 & x_1 & x_8 \\ x_3 & x_0 & x_7 \\ x_4 & x_5 & x_6 \end{array}$$

The plane $Z = ax + by + c$ should be fitted to the pixels under consideration. First consider a four pixel fit where the error e is given by

$$e = (ai + bj + c - x_0)^2 + (a(i-1) + bj + c - x_1)^2$$
$$+ (a(i-1) + b(j-1) + c - x_2)^2 + (ai + b(j-1) + c - x_3)^2.$$

Minimizing e with respect to a, b with c = 0 gives

$$a = \frac{x_0 + x_3}{2} - \frac{x_1 + x_2}{2}$$

$$b = \frac{x_0 + x_1}{2} - \frac{x_2 + x_3}{2}$$

If the plane $Z = lx + my + n$ were fit (using the same type of error criteria) to all nine pixels after minimization, we obtain

$$l = \frac{1}{2}\left[\frac{x_4 + x_5 + x_6}{3} - \frac{x_1 + x_2 + x_8}{3}\right]$$

$$m = \frac{1}{2}\left[\frac{x_6 + x_7 + x_8}{3} - \frac{x_2 + x_3 + x_4}{3}\right]$$

A weighted fit could also be used. Specifically, we could find the plane

$$Z = px + qy + r$$

which is best with respect to the same criteria as above, but weighs the grey values x_1, x_3, x_5, x_7 by a fact : of two. A gradient is said to exist when some functional (specified next) of a and b, or l and m, or p and q exceeds a threshold value. The most commonly employed functions are:

the l_1 norm

$$(A = |a| + |b|,\ B = |l| + |m|,\ \text{or}\ C = |p| + |q|)$$

the l_2 norm

$$(D = \sqrt{a^2 + b^2},\ E = \sqrt{l^2 + m^2},\ \text{or}\ F = \sqrt{p^2 + q^2})$$

the l_∞ norm

$$(G = \max(|a|,|b|),\ H = \max(|l|,|m|),\ I = \max(|p|,|q|))$$

Various versions of the Roberts, Prewitt and Sobel gradient result by application of these different norms. This is illustrated in the table below, using G_1 and G_2, as defined for various templates.

For the Roberts Template

$$G_1 = x_0 - x_2 \text{ and } G_2 = x_1 - x_3$$

For the Prewitt and Sobel templates

$$G_1 = \frac{1}{2+w}(x_4 + wx_5 + x_6) - (x_2 + wx_1 + x_8)$$

$$G_2 = \frac{1}{2+w}(x_6 + wx_7 + x_8) - (x_2 + wx_3 + x_4)$$

with w = 1 and w = 2, respectively (Table D-1).

Table D-1. Gradient Operators and Their Norm

RMS Criteria	Max Criteria	Magnitude
Roberts: $D = \frac{1}{\sqrt{2}} \sqrt{G_1^2+G_2^2}$	$H = \max(\lvert G_1\rvert, \lvert G_2\rvert)$ $= \lvert a\rvert + \lvert b\rvert$	$G = \frac{1}{2}(\lvert G_1\rvert+\lvert G_2\rvert)$
Prewitts: $E = \frac{1}{2} \sqrt{G_1^2+G_2^2}$	$H = \frac{1}{2}\max(\lvert G_1\rvert, \lvert G_2\rvert)$	$G = \frac{1}{2}(\lvert G_1\rvert+\lvert G_2\rvert)$
Sobels: $F = \frac{2}{3} \sqrt{G_1^2+G_2^2}$	$I = \frac{2}{3}\max(\lvert G_1\rvert, \lvert G_2\rvert)$	$C = \frac{2}{3}(\lvert G_1\rvert+\lvert G_2\rvert)$

Operators for this algebra involve the basis operations previously described herein. The side conditions that the operations obey follow in a natural manner, since all operators are range or domain induced. However, this listing will be given as part of the Phase II effort.

b. Dilate (Black and White)

(1) Classification:

Primitive Operation for Texture Analysis and Feature Generation.

(2) Purpose and Methodology:

Dilation is the dual morphological operation of erosion.

Whereas erosion is a shrinking transformation, dilation is expanding.

(3) Mathematical Description:

Let $X \subset ZxZ$ and define
$D_B(x) = \{(Z_1, Z_2) \in ZxZ: \; (B + (Z_1, Z_2)) \cap X \neq \emptyset$

If we define the Minkowski Addition as

$$X \oplus B = (X^c \ominus B)^c$$

where X^c denotes the complement of X, then we. have

$$D_B(X) = X \oplus B$$

It is important to note that dilation is the dual of erosion in the sense that:

$$D_B(X) = [E_B(X^c)]^c$$

(4) Transformation Type:

Image to Image
Increasing
Invariant under Translation

(5) Effectiveness and Deficiencies:

Comments analogous to those for erosion can be made. In particular, for the Euclidean counterpart: If Ψ is an increasing, translation invariant mapping on the power set of RxR, then Ψ is an intersection of dilations.

In fact,

$$\Psi(A) = \bigcap_{B \in V^*} A \oplus \hat{B}$$

where V^* is the kernel of the dual mapping Ψ^*, which is defined by

$$\Psi^*(A) = [\Psi(A^c)]^c$$

Note that, although we have quoted this theorem of Matheron, together with its erosion counterpart, for Euclidean images, corresponding results do hold for ZxZ.

(6) Alternate:

Miller's Expand Transformation

(7) References:

Serra, p. 43.
Matheron, pp. 17, 221.
Miller, p. 16.
Watson, p. 4.

c. Discrete Fourier Transform

(1) Classification:

Image Transformation

(2) Purpose and Methodology:

The Discrete Fourier Transform (DFT) is one of the most used transformations on images and is utilized in almost all facets of imaging, such as image restoration, enhancement, segmentation, etc. It is often employed as an approximation to the actual Fourier Transform operation for continuous images. It is described herewithin as an exact transform operating on an image f given as an M by N matrix of complex numbers. In particular, it is specified below as a special type of Discrete Picture Transform.

(3) Mathematical Description:

In the context of Discrete Picture Transform D, D: $A \rightarrow A$ with $D(f) = F = P \bullet f \bullet Q$. The DFT F of f is obtained if one uses $P = L_{MM}$ and $Q = L_{NN}$ with $L_{JJ} = \frac{1}{J}e^{-i\frac{2\pi}{J}}$ m,n = 0,1,2,...,J-1.

The inverse operator $D^{-1}(F) = f$ is called the inverse DFT. An element of F is given by

$$F(u,v) = \frac{1}{MN}\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n)\ e^{-[i\pi(\frac{mu}{M} + \frac{nv}{N})]}$$

$u = 0,1,2,\ldots,M-1$ and $v=0,1,2,\ldots,N-1$.

The function F can be extended to $\overline{F}$, which is doubly periodic and defined over ZxZ, and is such that

$$\overline{F}(u,-v) = F(u,N-v)$$

$$\overline{F}(-u,v) = F(M-u,v)$$

$$\overline{F}(-u,-v) = F(M-u,N-v)$$

or, more succinctly

$$\overline{F}(aM + u, bN + v) = F(u,v)$$

$a,b \in Z$. Using the fact that P and Q are invertible, similar properties on the original image f can be derived. The periodic extension property described above is one key in deriving important consequences of the DFT for convolving images.

(4) Transformation Type:

Image to image

(5) Effectiveness and Deficiencies:

The DFT is one of the key transform techniques in digital image processing; however, a large amount of computation must be performed to perform the transformation. This is because complex values and exponentials are needed. A further shortcoming is that when the DFT is employed as a numerical approximation to the True Fourier Transform. **(Some type of error or bounds on the** error must be registered.)

(6) Alternate Versions:

Various fast versions of the DFT exist under the global name Fast Fourier Transform of FFT. Some transforms exist which provide the same or similar results as the DFT, along with error bounds for use in approximation. One such algorithm is the Accurate Fourier Transform AFT.

(7) References:

E. Hall, pp. 123-138.
A. Rosenfeld and A. Kak pp. 20-24.
R. Bracewell pp. 376-384.

d. Discrete Picture Transform

(1) Classification:

General Linear Image Transformation Schema

(2) Purpose and Methodology:

The purpose of this general transform is to provide a global common setting for numerous important image processing transforms. The Discrete Picture Transform is defined for image f, given by a complex valued M x N matrix where

$$f = \begin{pmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & & & \\ \cdot & & & \\ \cdot & & & \\ \cdot & & & \\ f(M-1,0) & \dots & & f(M-1,N-1) \end{pmatrix}$$

Let A denote the set of all such matrices. The entries of this matrix are often real and denote grey values of f at designated points. The transform is defined as a linear operation on images.

(3) Mathematical Description:

The Discrete Picture Transform D is defined by: $D : A \longrightarrow A$, where $D(f) = F = P \bullet f \bullet Q$, where P and Q are nonsingular, complex-valued M x M and N x N matrices respectively, and are not functions of the image being transformed. Specific transforms arise by the way values are given for P and Q. The transforms are often called separable, since P operates on the columns and Q on the rows of f. The entries of the M x N matrix F are given by

$$F(u,v) = \sum_{M=0}^{M-1} \sum_{r=0}^{N-1} P(u,m)\, f(m,n)\, Q(n,v)$$

$$u = 0,1,\ldots,M-1$$
$$v = 0,1,\ldots,N-1$$

By pre-multiplying $F = P \bullet f \bullet Q$ on the left by the inverse matrix P^{-1} of P and post multiplying the expression by Q^{-1}, we obtain

$$f = P^{-1}\, F\, Q^{-1}$$

We call this transform which takes the image F back into f the Inverse Picture Transform and denote this by

$$D^{-1} : \; A \rightarrow A \text{ where } D^{-1}\,(F) = P^{-1}\,(F)\,Q^{-1}$$

(4) Transform Type:

Image to Image.

(5) Effectiveness and Deficiencies:

Numeróus Image transform techniques, such as the Hadamard or the Discrete Fourier Transform are representable, using this schema. The Discrete Picture Transform is a Linear Operation, viz, for any f_1, f_2in A and complex number α, $D(\alpha f_1 + f_2) = \alpha D(f_1) + D(f_2)$. As a consequence, nonlinear operations on images cannot be employed using this schema.

(6) Alternate Versions:

Matrices could be labeled using different indices. More importantly, the Discrete Picture Transform could be viewed as a special case of more general nonlinear transforms. An instance would be the Affine Transform

$$a, \text{ where } a: A \rightarrow A \text{ and } (f) = P \bullet f \bullet Q + Q_0 = F$$
$$\text{for some } Q_0 \text{ in } A.$$

Here, the Inverse Affine Transform is such that

$$a^{-1} : A \rightarrow A \text{ and } a^{-1} (F) = P^{-1}[(F) - Q_0]Q^{-1}$$

(7) References:

A. Rosenfeld and A. Kak pp. 19-20.

e. Erosion (Black and White)

(1) Classification:

Primitive Operation for Texture Analysis and Feature Generation.

(2) Purpose and Methodology:

Erosion is one of the two basic morphological operations. Its power lies in the multitude of higher level operations it generates by the use of different structuring elements and by iteration. Essentially, it operates by fitting small structuring elements, usually convex, into a given black and white figure.

(3) Mathematical Description:

An image $F: ZxZ \rightarrow \{0, 1\}$ is equivalent to a set $X \subset ZxZ$, i.e. F is the characteristic function of X For $B \subset ZxZ$, we define $E_B(X) = \{(Z1,Z2) \in ZxZ: B + (Z1,Z2) \subset X\}$. If we define the Minkowski Subtraction as

$$X \ominus B = \bigcap_{b \in B} X + b$$

then we have

$$E_B(X) = X \ominus \hat{B}$$

where

$$\hat{B} = \{ -b : b \in B\}$$

(4) Transformation Type:

Image to Image
Increasing
Invariant under Translation

(5) Effectiveness and Deficiencies:

In the present form, the transformation is limited to black and white images. Nonetheless, its power is significant. To emphasize its power, we note a result from its Euclidean counterpart, which operates on images R x R → {0,1}. If Ψ is an increasing, translation invariant mapping on the power set of R x R, then Ψ is a union of erosions. In particular

$$\Psi(A) = \bigcup_{B \in V} A \ominus \hat{B}$$

where

$$V = \{X \subset R x R : 0 \in \Psi(x)\}$$

V being called the kernel of Ψ.

In practice, successive erosions, each followed by a measurement, generate morphological feature criteria which quantitatively describe textural aspects of the image.

(6) Alternate:

(a) Miller defines the Shrink Transformation in terms of his general neighborhood transformations.

(b) Erosion operators can be defined for discrete level grey-tone images.

(7) References:

Serra, p. 43.
Matheron, pp. 17, 221.
Miller, p. 13.
Watson, p. 6.

f. Hadamard Transform (Walsh Transform)

(1) Classification:

Image Transformation

(2) Purpose & Methodology;

The Hadamard Transform is employed in numerous areas of image processing, such as image restoration, enhancement, compression, segmentation, classification, etc. It is defined below as a special case of the Discrete Picture Transform.

(3) Mathematical Description:

Consider the Discrete Picture Transform D, D: A →A, where

$D(f) = F = P \bullet f \bullet Q$, and P and Q are Hadamard matrices. Then D is called a Hadamard Transform. A Hadamard matrix $H_{JJ}$is a symmetric J x J matrix consisting of 1's and -1's, such that all rows (columns) are mutually orthogonal, using the Euclidean inner or dot product. The values of J employed here will be a power of 2, i.e., $J = 2^n$ for n = 1, 2,.... Furthermore, it is known that if a Hadamard matrix of rank $n > 2$ exists, then $n = 4m$, where m is an integer. Thus, the first interesting Hadamard matrix is H_{22}, where

$$H_{dd} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

By using the Theorem: If H_{JJ} is Hadamard, then

$$H_{2J2J} = \begin{pmatrix} H_{JJ} & H_{JJ} \\ H_{JJ} & -H_{JJ} \end{pmatrix}$$

is Hadamard, numerous other Hadamard matrices can be found. From $F = H_{MM}\ f\ H_{NN}$, we have the inverse relation

$$f = \frac{1}{MN} H_{MM}\ F H_{NN}$$

(4) Transformation Type:

Image to Image

(5) Effectiveness and Deficiencies:

Due to the nature of the P and Q matrices, no multiples are needed in determining the Hadamard Transform; only additions and subtractions of the grey values must be employed. This is a very computationally efficient transform.

(6) Alternate Versions:

There are faster versions of the Hadamard Transform. The Walsh Functions can be used to provide a transform equivalent to the Hadamard Transform.

(7) References:

A. Rosenfeld and A. Kak pp. 24-28.

g. Hotelling Transform Karhunen-Loeve Transform

(1) Classification:

Image Coding

(2) Purpose and Methodology:

This image processing procedure is useful in numerous applications of image processing, e.g., image compression, restoration, enhancement, rotation, feature selection, etc. All these applications are based on minimum variance estimation criteria employed in deriving the Hotelling Transform. The Hotelling Transform is a nonlinear operation on images and, as a consequence, is not a Discrete Picture Transform. Consider an f (i.e. a M by N matrix of reals)

$$f = \begin{pmatrix} f_1' \\ f_2' \\ \cdot \\ \cdot \\ \cdot \\ f_N' \end{pmatrix}$$

where f_i' is a 1 by N vector denoting the ith row of f, and the prime denotes transpose. Represent this image as an M·N by 1 column vector x, i.e.,

$$x = \begin{pmatrix} f_1 \\ f_2 \\ . \\ . \\ . \\ f_M \end{pmatrix}$$

The Hotelling Transform of x (or, equivalently, of the image f represented as a linear list x) is

$$y = Ax + Q_0$$

where A is a J by N•M matrix described below, and Q_0 is a 1 vector 1 $\leq J \leq NM$. This transform appears to be Affine; however, this, too, is not the case, for in practice A and Q_0 are complicated functions of x (or f). The matrices A and Q_0 involve moments from x where x is viewed as a random vector.

(3) Mathematical Description:

Consider K ($\geq$ 1) M by N real valued images $f_1, f_2, \ldots, f_k$. Let the K column vectors $x_1, x_2, \ldots, x_k$ be the linear list representation of these images as described above. Assume that x is an M N by 1 random vector and the probability that x equals x_i, is 1/K, i.e. $P(x = x_i) = 1/K$, $i = 1,2,\ldots,K$. Thus all the x_i are equilikely realizations of x. As usual, let $\bar{x}$ denote the average or mean vector and so

$$\bar{x} = \frac{1}{K} \sum_{i=1}^{K} x_i$$

Also, let R be the covariance matrix for the process

$$R = \frac{1}{K} \sum_{i=1}^{K} (X_i - \bar{x}) \; (x_i - \bar{x})'$$

We will assume that R is full rank. Since R is symmetric and real, all the eigenvalues are real and there is a internormal basis of associated eigenvectors. Let A_J be the matrix whose rows consist of the normalized eigenvectors corresponding to the J largest eigenvalues. The integer J is a parameter here and different Hotelling Transforms are defined as a consequence. We have as the Hotelling Transform

$$y_J = A_J(x - \bar{x}) = A_J x - A_J \bar{x}$$

It should be noted that y_J is a random vector; furthermore, for

$J = N \cdot M$, we use $A_{NM} = A$ and $y_J = y$

Thus, in this case, we have

$$y = Ax - A\bar{x}$$

This M•N by 1 vector y could be interpreted as an image g. To see this, we represent y as a M by N vector

$$y = \begin{pmatrix} f_1 \\ f_2 \\ . \\ . \\ . \\ f_m \end{pmatrix}$$

where each g_i is an N x 1 column vector. Then the image g is given by

$$g = \begin{pmatrix} g_1' \\ g_2 \\ . \\ . \\ . \\ g_m' \end{pmatrix}$$

(4) Transformation Type:

Image to Image
Image to Vector

(5) Effectiveness and Deficiencies:

The Hotelling Transform utilizes orthonormal bases and therefore is distance preserving. It is derived by minimizing a mean square type error, and, therefore, it is optimal under this cost function. The calculations involved in producing these transforms are computationally complex. The procedure described above was exact. Often one performs this procedure using $x_1, x_2, \ldots, x_k$ as samples from a (larger) population. In this case

$$\bar{x} = E(x)$$

and

$$R = E[(x - \bar{x}) \quad (x - \bar{x}' \quad) \}$$

should be employed, and the values of $\bar{x}$ and R given in the Mathematical Description are statistics and therefore only approximations to the true parameters $\bar{x}$ and R.

(6) Alternate Versions:

Numerous versions exist, although none which are fast. They are recognized under the following names: Discrete Karhunen-Loeve Transforms, Principal Component Transforms and Eigenvector Transforms. Some of these methods do not utilitize the term $Q_0 = A_J \bar{x}$

(7) References:

M. Kendall and A Stuart pp. 292-323.

E. Hall, pp.115-122.

J. Tow, R. Gonzalez, pp. 271-283.

h. Opening (Black and White)

(1) Classification:

Feature Generation and Filtering, Size and Shape Description.

(2) Purpose and Methodology:

The opening is essentially a fitting operation. The opening of a domain X is the region swept out by the translates of the structuring element B. It smooths the contours of X, eliminates negligible components, and suppresses narrow dendritic extensions. Iterations of openings play a crucial role in generating size and shape descriptors.

(3) Mathematical Description:

For a structuring element B and $X \subset Z{\times}Z$, we define the opening of X by B to be:

$$X_B = (X \ominus \hat{B}) \oplus B.$$

Equivalently,

$$X_B = [B + y : B + y \subset X]$$

(4) Transformation Type:

Image to Image

Increasing,

Anti-extensive: $X_B \subset X$

Idempotent: $(X_B)_B = X_B$

(5) Effectiveness and Deficiencies:

If we consider the typical morphological feature description operation,

$$\text{Image} \rightarrow \text{Image} \rightarrow \text{Parameter},$$

the application of successive openings at the Image → Image stage is determined by the function of the parametric measurement. For example, if we open by ever larger sets of a particular class, more and more resolution of the micro-texture will be filtered. It is precisely the measurement of the filtered micro-texture which is descriptive of size and shape. The effectiveness of the opening, or its non-effectiveness, is thereby determined, at least insofar as any particular application is concerned.

It should be noted that openings characterize an important class of morphological mappings (on Euclidean Images): If Ψ is translation invariant, increasing, anti-extensive and idempotent,

$$\Psi: p(R \times R) \rightarrow p(R \times R),$$

then there exists a class $B_0 \subset p(R \times R)$ such that

$$\Psi(A) = \bigcup \{A_B : B \in B_0\}$$

and conversely, where we let $p(R \times R)$ denote the power set.

(6) Alternate:

(7) References:

Serra, p. 50.

Matherson, pp. 18, 190.
Watson, p. 6.

i. Size Criteria

(1) Classification:

Size (and ipso facto shape) criteria-general analysis.

(2) Purpose and Methodology:

The purpose of any given size criteria is to create a distribution associated with the image which reflects its size and shape characteristics, from a textural level. The methodology is as follows:

(a) Every operator Λ will denote a parametized family of operators Λ_λ, each of the following form:

$$\Lambda_\lambda = \mu \bullet \Psi_\lambda, \; \lambda > 0$$

where

$$\{0,1\}^{Z\times Z} \xrightarrow{\Psi_\lambda} \{0,1\}^{Z\times Z} \xrightarrow{\mu} R$$

The family $\{\Psi_\lambda\}$ will be a granulometry (see (b)) and μ will usually be a Minkowski functional.

(b) A granulometry $\{\Psi_\lambda\}$ on the power set of Z x Z must satisfy:

(1) $\Psi_\lambda(A) \subset A$ for all A
(2) $A \subset B \Rightarrow \Psi_\lambda(A) \subset \Psi_\lambda(B)$
(3) $\Psi_\lambda \bullet \Psi_\mu = \Psi_{\sup(\lambda,\mu)} \; \lambda,\mu > 0$

The following theorem of Matheron is important in this regard:

Ψ_λ, $\lambda > 0$ is a granulometry iff

1. $\forall \lambda > 0, \Psi_\lambda$ is increasing, idempotent and anti-extensive, and,
2. $\lambda \geq \mu > 0$ implies $B_\lambda \subset B_\mu$, where B_λ denotes the class of Ψ_λ.

3. In morphology, one usually digitalizes by utilizing a hexagonal grid. It is important to keep this in mind when discussing directions. Of course, the digital theory is applicable to Z x Z in general, and, hence square grids in particular.

(3) Mathematical Description:

Depends on particular granulometry and particular measure.

(4) Transformation Type:

Image to Real
Image to Distribution

(5) Effectiveness and Deficiencies:

Effectiveness depends upon particular operator and the goal desired. In all cases, the choice of the structuring element is crucial. Moreover, interpretation depends upon expertise. It is here when an expert system would be of fundamental importance.

(6) Alternate:

(7) Reference:

Matheron, pp. 24, 192.

j. **Thresholding**

(1) Classification:

Segmentation

(2) Purpose and Methodology:

In thresholding, a figure, or an interesting feature, within an image is separated out from the image by the production of a new image in which the figure is black and its background is white. Variants of this methodology consist of whiting out or blackening only portions of the image, while leaving other grey levels intact.

(3) Mathematical Description:

In general, a thresholded image X results from an original image X by:

$$X_{\Lambda}(i,j) = \begin{cases} 1 & \text{if } X(i,j) \in \Lambda \\ 0 & \text{if } X(i,j) \notin \Lambda \end{cases}$$

where Λ is some subset of the grey scale. In particular, thresholding is usually defined in the case where

$$\Lambda = \{ Z : Z \geq \lambda \}$$

In this instance, we defined

$$X_{\lambda}(i,j) = \begin{cases} 0 & \text{if } X(i,j) < \lambda \\ 1 & \text{if } X(i,j) \geq \lambda \end{cases}$$

Whereas, in thresholding proper,

$$X \in R^{2x2} \rightarrow X_{\wedge} \notin \{0,1\}^{2x2}$$

in semi-thresholding,

$$X \in R^{2x2} \rightarrow X_{\wedge} \notin R^{2x2}$$

where the actual portion of the grey-scale (^) utilized is reduced via

$$X_{\lambda}(i,j) = \begin{cases} x(i,j) & \text{if } X(i,j) \in \wedge \\ 0 & \text{if } X(i,j) \notin \wedge \end{cases}$$

For a simple threshold, λ, selection is crucial if the object is to be clearly delineated. Very often a strongly bimodal histogram indicates a sharp figure and ground distinction; however, this is certainly not always the case. Therefore, the typical method of choosing between the modes must be determined judiciously.

(4) Transformation/Type:

Image to Image
Usually: $R^{2x2} \rightarrow \{0,1\}^{2x2}$

(5) Effectiveness and Deficiencies:

As has been hinted above, figure extraction by bimodal thresholding is problematic. One need only consider the instances of shadows or high frequency salt-and-pepper noise. In many instances, feature or figure extraction may require pre-processing; i.e.,

$$X \in R^{Z \times Z} \rightarrow \hat{X} \in R^{Z \times Z} \rightarrow X_{\lambda} \in \{0,1\}^{Z \times Z}$$

A typical instance might be noise reduction by smoothing, followed by thresholding to separate the figure from the ground. In any event, the choice of threshold is the most crucial aspect of thresholding and several methods, including bimodal selection, are available. Should the threshold value λ be too small or too high, a noisy image will likely result.

One possibility for threshold selection is local thresholding. For example, one portion of an image might be lighter than another. In this case, no single value for λ would do. It may be possible to partition the image, threshold locally, and then smooth the resulting thresholded and partitioned image. Note that, in this instance, the smoothing simply obviates false edges that occur along partition boundaries; it does not visually smooth since the thresholded image may be black and white.

(6) Alternates:

(a) Minimum Error Thresholding
(b) Variable Thresholding

(7) References:

Rosenfeld, pp. 258-269.
Pavlidis, p. 66.
Serra, pp. 433-457.

APPENDIX E

A BRIEF DISCUSSION OF MANY SORTED ALGEBRA

A many sorted algebra is a very general algebraic structure. It is a generalization of a Universal Algebra and, consequently, it is a super structure for groups, rings, integral domains, lattices, and all other structures definable within the Universal Algebra framework. In short, a Universal Algebra consists of three ingredients. The first is a single non-empty set of elements; the second is various operators which map elements from these sets into other elements in this set, and the last is a collection of side conditions or equational constraints, such as the commutative, associative, or distributive laws, which the operators obey. In a many-sorted algebra, more than one type of set of elements is allowed. The operators in this type of algebra map elements from numerous sorts of sets into an element in some sort of set. A variety specification is also allowed, using equational constraints.

A most elegant and basic example of a many-sorted algebra is a vector space. Here, there are two sorts of sets, namely, vectors and scalars. Numerous operators, such as vector addition and multiplication of a vector by a scalar, are among the operators in this algebra. Furthermore, various side conditions such as the commutative law for addition and various distributive laws are well-known for a vector space structure.

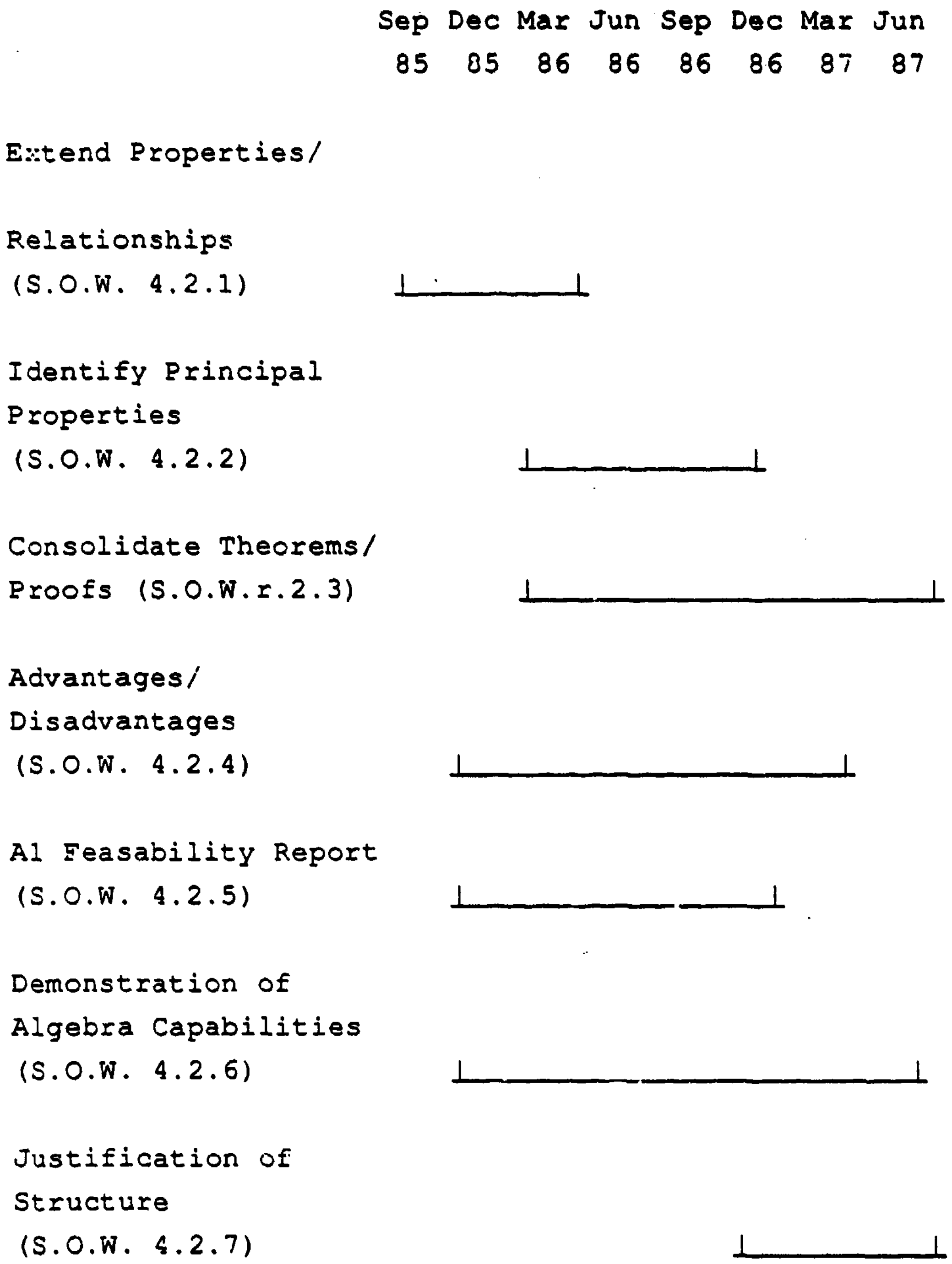
The application at hand, namely the imaging algebra being proposed, is a special type of many sorted algebra. Among the sort of sets involved are images, reals, and integers. The operators for this algebra involve the basis operations discussed in paragraph 2 and are described herein. The side conditions that the operations obey follow in a natural manner, since all operators are range or domain induced. However, this listing will be given as part of the Phase II effort.

APPENDIX F

SCHEDULE OF PHASE II DEVELOPMENT TASKS

* * * * *

TENTATIVE SCHEDULE OF PHASE II MILESTONES



REFERENCES

1. Mr Matheron, Radom Sets and Integral Geometry, John Wiley, 1975.

2. Mr Miller, An Investigation of Boolean Image Neighborhood Transformation, Doctoral Dissertation, Ohio State University, 1978.

3. Mr Watson, Mathematical Morphology, Tech Report 21, Series 2, Department of Statistics, Princeton University, March 1973.

4. Mr Bracewell, The Fourier Transform and Applications, 2nd Edition, McGraw Hill, 1978.

5. Mr M. Kendall and Mr A. Stuart, Theory of Statistics, 3rd Edition, Hafner Press, 1975.

6. Mr J. Tow and Mr R. Gonzalez, Pattern Recognition Principles, Addison Welsey, 1974.

BIBLIOGRAPHY

Aggarwal, J.K., R. O. Duda, and A. Rosenfeld, eds. Computer Methods in Image Analysis. New York: IEEE Press, 1977.

Aplin, G. J., and T. I. Binford. (1973) "Computer Description of Curved Objects," Proc. of the Intern. Joint Conf. on Artificial Intelligence, Sanford, California (August 20-23, 1973):629-640.

Ahuja, N., and B.J. Schachter. Pattern Models, New York: John Wiley & Sons, 1983.

Andrews, H.C. Computer Techniques in Image Processing. New York: Academic Press, 1970.

Andrews, H.C., ed. Digital Image Processing. New York: IEEE Press, 1978.

Arcelli, C. "Pattern Thinning by Contour Tracing," Computer Graphics and Image Processing, Vol. 17, No. 3 (October 1981): 130-144.

Bajcsy, R. "Computer Identification of Visual Surfaces," Computer Graphics and Image Processing, Vol. 2 (October 1973): 118-130.

Bajcsy, R. "Three-Dimensional Scene Analysis," Pro. Pattern Recognition Conf., Miami, Fla. (December 1-4, 1980): 1064-1074.

Ballard, D.H., and C. M. Brown, Computer Vision. Englewood Cliffs, N.J.: Prentice-Hall, Inc. 1982.

Barrow, H.G., and J. M. Tenebaum. "Recovering Intrinsic Scene Characteristics from Images," Computer Vision Systems, A. R. Hanson and E. M. Riseman, eds. New York: Academic Press, 1978.

Barrow, H.G., and J. M. Tenebaum, "Computational Vision," Proc of the IEEE. Vol. 69, No. 5 (May 1981): 572-595.

Bernstein, R., ed. Digital Image Processing for Remote Sensing. New York: IEEE Press, 1978.

Binford, T.O. "Visual Perception by Computer," Proc. IEEE Systems Science and Cybernetics Conf. Miami (December 1971).

Blahut, R.E. Fast Algorithms for Digital Signal Processing. Reading, MA: Addison-Wesley, 1985.

Brady, J.M., ed. Computer Vision. Amsterdam: North-Holland Publishing Co., 1981.

Brice, C.R., and C. L. Fennema. "Scene Analysis Using Regions," Artifical Intelligence, Vol. 1, No. 3 (Fall 1970): 205-226.

Canny, J. "Finding Edges and Lines in Images," MIT AI Laboratory Technical Report 720 (June 1983).

Castleman, K.R. Digital Image Processing. Englewood Cliffs, NJ: Prentice-Hall, Inc. 1979.

Cornsweet, T.N. Visual Perception. New York: Academic Press, 1970.

Davis, L.S. "A Survey of Edge Detection Techniques," Computer Graphics and Image Processing, Vol. 4 No. 3 (September 1975): 248-270.

Dodd, G. G., and L. Rossol, eds. Computer Vision and Sensor-Based Robots. New York: Plenum Press, 1979.

Duda, R.O., and P. E. Hart. Pattern Classification and Scene Analysis. New York: John Wiley & Sons, 1973.

Faugeras, O. D., ed. Fundamentals in Computer Vision. Cambridge: Cambridge Univ. Press, 1983.

Freeman, H. "Techniques for the Digital Computer Analysi of Chain-Encoded Arbitrary Plane Curves," Proc. National Electronics Conf. Vol. 17 (Oct 9-11, 1960): 421-432.

Freuder, E.C. "On the Knowledge Required to Label a Picture Graph," Artificial Intelligence Vol. 15, Nos. 1 & 2 (November 1980): 1-17.

Gardner, W.E., ed. Machine-Aided Image Analysis, 1978. Bristol & London: The Institute of Physics, 1979.

Gonzalez, R.C., and P. Wintz. Digital Image Processing. Reading, MA: Addison-Wesley, 1977.

Green, W.B. Digital Image Processing: A Systems Approach. New York: Van Nostrand Reinhold Co., 1983.

Gupta, J.N., and P. A. Wintz. "A Boundary Finding Algorithm and Its Application," IEEE Trans. on Circuits and Systems, Vol. 22, No. 4 (April 1975): 351-362.

Habibi, A. "Two Dimensional Bayesian Estimation of Images," Proc. of the IEEE, Vol. 60, No. 7 (July 1972): 878-883.

Hall, E. Computer Image Processing and Recognition. New York: Academic Press, 1979.

Hanson, A.R., and E. M. Riseman, eds. Computer Vision Systems. New York: Academic Press, 1977.

Haralick, R.M. "Edge and Region Analysis for Digital Image Data," Computer Graphics and Image Processing, Vol. 12, No. 1 (January 1980) : 60-73.

Herman, G.T., ed. Image Reconstruction From Projections - Implementation and Applications, New York: Springer-Verlag, 1979.

Hildreth, E. C. The Measurement of Visual Motion. Cambridge, MA: MIT Press, 1983.

Huang, T.S., ed. Image Sequence Processing and Dynamic Scene Analysis. New York: Springer-Verlag, 1983.

Huang, T.S., W. F. Schreiber, and O. J. Tretiak, "Image Processing" Proc. of the IEEE, Vol. 59, No. 11 (November 1971):1586-1609.

Hueckel, M. "An Operator Which Locates Edges in Digital Pictures," Journal of the ACM, Vol. 18, No. 1 (January 1971):113-125.

M. Hueckel, "A Local Visual Operator Which Recognizes Edges and Lines," Journal of the ACM, Vol. 20, No. 4 (October 1973):634-647.

Jacobus, C.J., and R. T. Chien. "Two New Edge Detectors," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 2, No. 5 (September 1981): 581-592.

Kanal, L.N., ed. Pattern Recognition, Washington, D.C.: Thompson Book Co., 1980.

Marr, D., and E. Hildreth. "Theory of Edge Detection," Proc of the Royal Society of London B, Vol. 207 (1980): 187-217.

Modestino, J.W., and R. W. Fries. "Edge Detection in Noisey Images Using Recursive Digital Filtering," Computer Graphics and Image Processing, Vol. 6, No. 5 (October 1977) 409-433.

Nevatia, R. Machine Perception. Englewood Cliffs, NJ: Prentice-Hall, Inc. 1982.

Norton, H.N. Sensor and Analyzer Handbook. Englewood Cliffs, NJ: Prentice-Hall, Inc. 1982.

Oppenheim, A.V., and A. S. Willsky. Signals and Systems. Englewood Cliffs, NJ: Prentice- Hall, Inc. 1983.

Pavlidis, T. "An Asynchronous Thinning Algorithm," Computer Graphics and Image Processing, Vol. 20, No. 2 (October 1982): 133-157.

Pratt, W. Digital Image Processing. New York: John Wiley

& Sons, 1978.

Rosenfeld, A., ed. Digital Picture Analysis. New York: Springer-Verlag, 1976.

Rosenfeld, A., and A. C. Kak. Digital Picture Processing. Vols 1&2, 2d ed. New York: Academic Press, 1982.

Rosenfeld, A. "Connectivity in Digital Pictures," Journal of the ACM, Vol. 17, No. 1 (January 1970): 146-160.

Schalkoff, R.J., and E. S. McVey. "A Model and Tracking Algorithm for a Class of Video Targets," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 4, No. 1 (January 1982) :2-10.

Scharf, D. Magnifications - Photography with the Scanning Electron Microscope. New York: Schocken Books, 1977.

Serra, J. Image Analysis and Mathematical Morphology. Boston, MA: Academic Press, 1985.

Shafer, S.A. Shadows and Silhouettes in Computer Vision. Boston, MA:Academic Press, 1985.

Stoffel, J.C., ed. Graphical and Binary Image Processing and Applications. Massachusetts: Artech House, Inc. 1982.

Stucki, P., ed. Advances in Digital Image Processing: Theory, Application, Implementation. New York: Plenum Press, 1979.

Sugihara, K. "Mathematical Structures of Line Drawings of Polyhedrons - Toward Man-Machine Communication by Means of Line Drawings," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 4, No. 5 (September 1982): 458-469.

Tanimoto, S., and A. Klinger, eds. Structured Computer Vision: Machine Perception through Hierarchical Computation Structures. New York: Academic Press, 1980.

Ullman, S. The Interpretation of Visual Motion. Cambridge, MA: MIT Press, 1979.

Ullman, S., and W. Richards, eds. Image Understanding. Norwood, NJ: Ablex Publishing Corp., 1984.

Wojcik, Z.M. "An Approach to the Recognition of Contours and Line-Shaped Objects," Computer Vision, Graphics and Image Processing, Vol. 25, No. 2 (February 1984): 184-204.

Woodham, R. J. "Analysing Images of Curved Surfaces," Artificial Intelligence, Vol. 17, Nos. 1-3 (August 1981): 117-140.